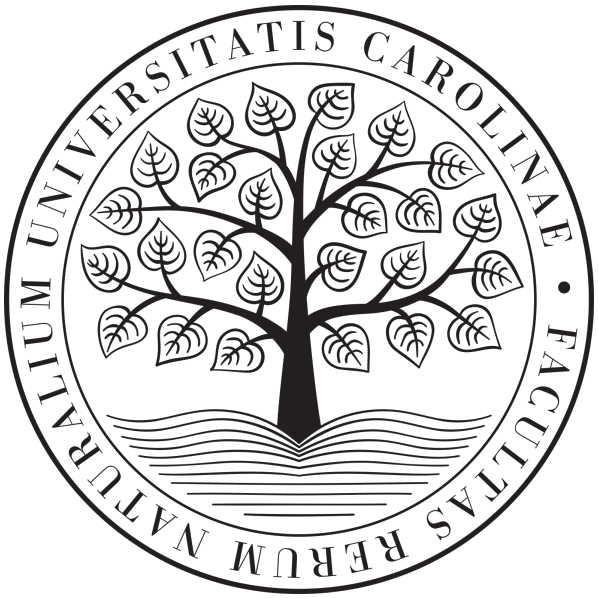
Univerzita Karlova

Přírodovědecká fakulta

Katedra aplikované geoinformatiky a kartografie

Studijní program: Geografie (magisterské studium)

Studijní obor: Kartografie a geoinformatika



Bc. Michal MATYÁŠ

KLASIFIKACE ZÁSTAVBY PRO ÚČELY KARTOGRAFICKÉ GENERALIZACE STÁTNÍHO MAPOVÉHO DÍLA

CLASSIFICATION OF BUILT-UP AREAS FOR CARTOGRAPHIC GENERALIZATION OF STATE MAP SERIES

Diplomová práce

Vedoucí diplomové práce: RNDr. Jakub Lysák, Ph.D.

Praha, 2020

Vysoká škola: Univerzita Karlova **Fakulta:** Přírodovědecká

Katedra: Aplikované geoinformatiky a kartografie **Školní rok:** 2019/2020

Zadání diplomové práce

pro Bc. Michala Matyáše

obor Kartografie a geoinformatika

Název tématu: Generalizace zástavby pro ZM 50 z dat ZABAGED

Zásady pro vypracování

Cílem diplomové práce bude navrhnout a otestovat algoritmus pro klasifikaci zástavby pro účely generalizace zástavby státního mapového díla v měřítku 1 : 50 000. Metoda bude navržena jak pro klasifikaci městské, tak i venkovské zástavby. Cílem bude prakticky otestovat a porovnat klasifikátory z oblasti strojového učení a neuronových sítí a zároveň porovnat klasifikátory využívající popisných charakteristik s klasifikátory využívajícími vizuální posouzení. Jako vstupní data budou použita data ZABAGED a Data50. Praktickým výstupem této práce bude polygonová vrstva shluků budov netriviálního rozsahu s přiřazeným typem zástavby.

Rozsah grafických prací: cca 15 stran

Rozsah průvodní zprávy: cca 70 stran

Seznam odborné literatury:

DU, S., LUO, L., CAO, K., SHU, M. (2016): *Extracting building patterns with multilevel graph partition and building grouping. ISPRS Journal of Photogrammetry and Remote Sensing, 122, 81–96.*

HE, X., ZHANG, X., XIN, Q. (2018*): Recognition of building group patterns in topographic maps based on graph partitioning and random forest. ISPRS Journal of Photogrammetry and Remote Sensing, 136, 26–40*.

YAN, X., AI, T., YANG, M., YIN, H. (2019): A graph convolutional neural network for classification of building patterns using spatial vector data. ISPRS Journal of Photogrammetry and Remote Sensing, 150, 259–273.

HECHT, R., HEROLD, H., MEINEL, G., BUCHROITHNER, M. (2013): *Automatic Derivation of Urban Structure Types from Topographic Maps by Means of Image Analysis and Machine Learning. In: 26th International Cartographic Conference, 2013, 18.*

Vedoucí diplomové práce: RNDr. Jakub Lysák, Ph.D.

Datum zadání diplomové práce: 19.12.2018

Termín odevzdání diplomové práce: jaro 2020

Platnost tohoto zadání je po dobu jednoho akademického roku.

…… ……

Vedoucí diplomové práce Vedoucí katedry

V Praze dne 20. prosince 2018

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně, pod vedením školitele RNDr. Jakuba Lysáka, Ph.D., a že jsem uvedl a řádně citoval všechny použité prameny.

Svoluji k zapůjčení této práce pro studijní účely a souhlasím s tím, aby byla řádně vedena v evidenci vypůjčovatelů.

Tato práce ani její podstatná část nebyla předložena k získání jiného nebo stejného akademického titulu.

V Praze, dne 15. května 2020 .....

Michal Matyáš

Poděkování

Na tomto místě bych rád upřímně poděkoval vedoucímu mé diplomové práce RNDr. Jakubovi Lysákovi, Ph.D., za čas, který mi věnoval, za jeho podnětné připomínky, rady a ochotu. Dále bych rád poděkoval Českému úřadu zeměměřickému a katastrálnímu, jmenovitě paní Růženě Chaloupecké za zapůjčení dat ze ZABAGED. V neposlední řadě děkuji také své rodině za podporu během studia a své přítelkyni, která mi byla oporou při psaní této práce.

Klasifikace zástavby pro účely kartografické generalizace SMD

Abstrakt

Tato diplomová práce se zabývá tématem automatické klasifikace zástavby. Hlavním cílem této diplomové práce bylo navrhnout algoritmus pro identifikaci typů zástavby pro účely kartografické generalizace. Pro účely této diplomové práce bylo vymezeno celkem šest typů zástavby s ohledem na rozdílnou generalizaci jednotlivých typů na ZM 50. První část navržené metody je představována algoritmem pro segmentaci budov do shluků založeného na využití již generalizované cestní sítě a algoritmu DBSCAN. Dílčím cílem této diplomové práce bylo porovnat klasifikátory z oblasti strojového učení a neuronových sítí a zároveň porovnat klasifikátory využívající popisných charakteristik s klasifikátory využívajících vizuální posouzení. Výsledné klasifikace byly zhodnoceny pomocí dat z ručně vybrané trénovací množiny a též pomocí algoritmu porovnávající výsledný typ zástavby s typem kartografické reprezentace využité pro znázornění zástavby na ZM 50. Celá metoda byla implementována v jazyce Python zejména s využitím knihoven Arcpy, Scikit-learn a Tensorflow. Testování probíhalo nad prvky z databází ZABAGED a Data50.

Klíčová slova: Generalizace zástavby, Klasifikace, Strojové učení, Neuronové sítě, ZABAGED, Data50

Classification of built-up areas for cartographic generalization of state map series

Abstract

This diploma thesis deals with the topic of automatic classification of buildings. The main goal of this diploma thesis was to design an algorithm for the identification of building types for the purposes of cartographic generalization. For the purposes of this diploma thesis, a total of six types of development were defined with respect to different generalization of individual types on ZM 50. The first part of the proposed method is represented by an algorithm for segmenting buildings into clusters based on the use of already generalized road network and DBSCAN algorithm. The partial goal of this diploma thesis was to compare classifiers from the field of machine learning and neural networks and at the same time to compare classifiers using descriptive characteristics with classifiers using visual assessment. The resulting classifications were evaluated using data from a manually selected training set and using an algorithm comparing the resulting type of development with the type of cartographic representation used to represent the development on ZM 50. The whole method was implemented in Python using Arcpy, Scikit-learn and Tensorflow libraries. Testing took place on elements from the ZABAGED and Data50 databases.

Keywords: Generalization of built-up areas, Classification, Machine learning, Neural networks, ZABAGED, Data50

Obsah

Seznam obrázků

Seznam tabulek

Seznam grafů

Seznam příloh

1 Úvod

Rozvoj digitálních technologií v kartografii a tlak na rychlejší aktualizace mapových podkladů vede v posledních letech k přechodu z ruční tvorby map na automatickou tvorbu. Již z definice mapy, jako zmenšeného zjednodušeného konvenčního obrazu Země, kosmu, kosmických těles nebo jejich částí převedeného do roviny pomocí matematicky definovaných vztahů (ČÚZK 2019), je patrné, že proces generalizace tvoří nedílnou součást tvorby každé mapy. Jednou ze součástí tvorby map středních měřítek je i generalizace zástavby. Mezi zásadní části procesu generalizace zástavby patří získání kontextuální informace neboli popis všech vztahů, které ovlivňují výslednou podobu. Právě automatizace získávání kontextuální informace patří v současné době k jednomu z nejvíce diskutovaných témat v oblasti automatické generalizace (Regnauld 2001).

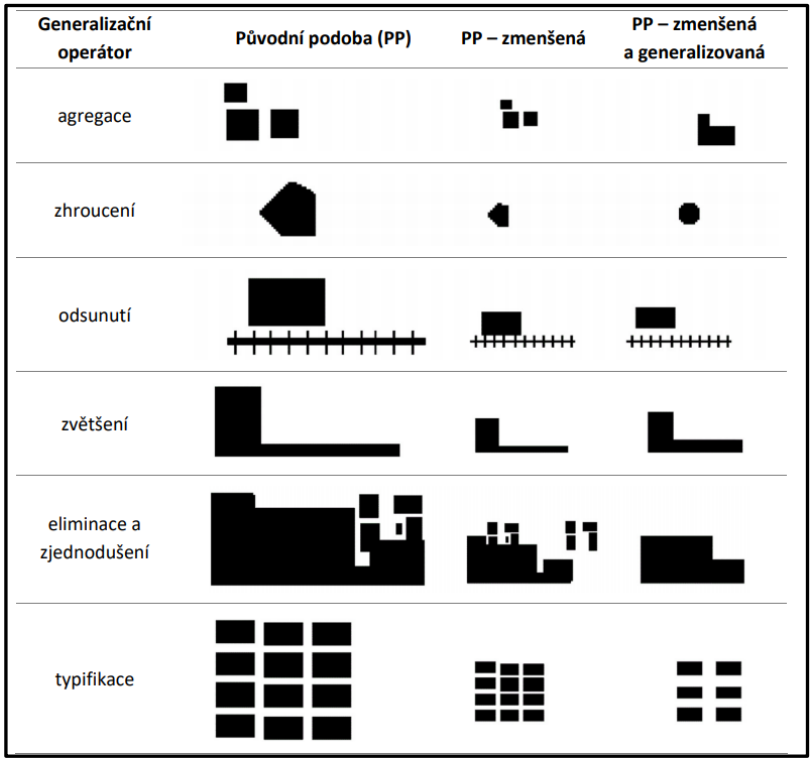
Cílem této diplomové práce je vytvořit algoritmus pro extrakci zmíněné kontextuální informace. Pro získání této informace je cílem klasifikovat zástavbu do několika typů, na které lze pak následně aplikovat odlišné generalizační operátory (algoritmy). Vlastní aplikace generalizačních operátorů již není předmětem této práce. Konkrétně je tato práce zaměřená na kategorizaci zástavby pro účely její generalizace na Základní mapě 1 : 50 000 (ZM 50). Pro splnění hlavního cíle práce byly vytyčeny tři dílčí cíle. Prvním dílčím cílem je představení typologie zástavby s důrazem na propojení jednotlivých typů a kartografických reprezentací využívaných pro znázornění zástavby na ZM 50. Druhým dílčím cílem práce je poté navržení algoritmu pro segmentaci budov do shluků. Posledním dílčím cílem práce je přiřazení představených typů zástavby jednotlivým shlukům budov neboli klasifikace shluků budov. Pro vlastní klasifikaci je cílem porovnat klasifikátory využívající popisných charakteristik a klasifikátory využívající vizuální posouzení. Cílem je též porovnat klasifikátory z oblasti strojového učení a neuronových sítí.

Celá diplomová práce je dělena do šesti kapitol. Druhá kapitola práce je věnována představení v současnosti využívaných algoritmů pro získávání kontextuální informace a bližšímu popisu fungování klasifikátorů využitých dále v práci. Ve třetí kapitole je představena navržená typologie zástavby včetně podrobné definice jednotlivých typů. Kapitola č. 4 je poté věnována popisu a implementaci navrženého segmentačního algoritmu a též procesu klasifikace s využitím jednotlivých klasifikátorů. Pátá kapitola je poté věnována srovnání jednotlivých klasifikátorů. V závěrečné šesté kapitole jsou

shrnuty výsledky dosažené v celé diplomové práci, včetně diskuse o splnění vytyčených cílů a možných vylepšení této diplomové práce.

2 Teoretická část

Cílem této kapitoly práce je poskytnout bližší pohled na problematiku generalizace zástavby pro mapy středních měřítek a zejména představit v současnosti využívané algoritmy pro získání kontextuální informace pro proces generalizace zástavby. V této kapitole práce jsou též blíže popsány základní principy fungování klasifikátorů využitých dále v práci.



Obrázek 1: Kontextuální generalizační operátory
***Zdroj**: Gottstein (2019)*

2.1 Generalizace zástavby

Úspěšná generalizace zástavby vyžaduje detailní znalosti vztahů a sil, které ovlivňují výslednou podobu generalizovaného objektu. Generalizace zohledňující všechny tyto faktory bývá nazývána jako **kontextuální kartografická generalizace** (Regnauld 2001). Při kontextuální generalizaci je zohledňován zejména tvar objektu, orientace, vzdálenost k jiným objektům, zarovnání a též sémantické informace. Proces vedoucí k získání kontextuální informace bývá nazýván jako **rozpoznávání vzorů** (*structure/pattern recognition*). Znalost struktury dat je nezbytným krokem pro aplikaci tzv. kontextuálních generalizačních operátorů (obrázek 1) (Li et al. 2004). Proces rozpoznávání vzorů pro generalizaci zástavby bývá obvykle dělen do dvou kroků (Basaraner, Selcuk 2008).

Prvním krokem je identifikace shluků budov. Cílem je nalézt vizuálně oddělitelné homogenní shluky budov v zástavbě. Důvod pro vymezení menších skupin zástavby vychází z Toblerova prvního zákona geografie („Všechno souvisí se vším, ale blízké věci spolu souvisejí více než věci vzdálené.“). Z tvarových zákonů Gestaltismu též vyplývá, že blízké objekty mají sklon být vnímány jako skupina. Jedním z dalších důvodů pro vymezení shluků zástavby je fakt, že blízko sousedící objekty mají větší pravděpodobnost grafických kontaktů (tj. porušení pravidla o minimální vzdálenosti objektů na mapě) ve výsledném měřítku mapy (Cetinkaya, Basaraner, Burghardt 2015). Druhým krokem je poté identifikace vzoru ve shluku budov neboli snaha o identifikaci typu zástavby každého shluku budov. Proces hledání vzorů je závislý na vymezené typologii zástavby.

Na typologii zástavby lze nahlížet z několika různých úhlů pohledu, v závislosti na oblasti výzkumu. Podle obytné funkce lze zástavbu rozdělit na rodinné domy, bytové domy a nebytové budovy (Meinel, Hecht, Herold 2009). Podle Steiniger et al. (2008) lze zástavbu klasifikovat na zástavbu v centrech měst, komerční a industriální zástavbu, obytnou zástavbu, suburbánní zástavbu a venkovskou zástavbu. S ohledem na tvorbu map středních měřítek je však nejdůležitější klasifikace zástavby z pohledu kartografické generalizace. V následující podkapitole je uveden přehled automatických metod využívaných pro segmentaci zástavby a též i několik různých typologií zástavby relevantních z pohledu kartografické generalizace.

2.2 Segmentace a klasifikace zástavby

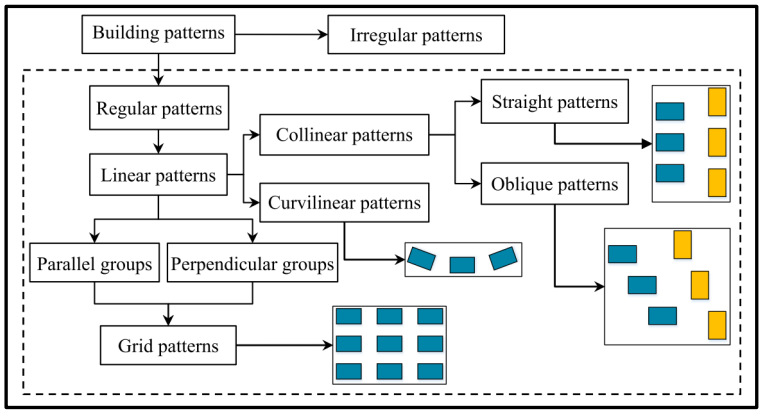
V odborné literatuře lze nalézt relativně velké množství publikací, které se zabývají **hledáním shluků budov** (*building clustering*). Většina těchto metod je založena na tvorbě grafů pro vyjádření sousednosti budov a výpočtu popisných charakteristik pro následné dělení těchto grafů pro nalezení shluků. Pro tento účel bylo popsáno mnoho podobnostních charakteristik, jako je například lokální hustota (Anders, Sester, Fritsch 1999). Yan, Weibel a Yang (2008) ve své práci pro vyhledávání shluků využili parametrů jako je minimální vzdálenost, poměr ploch, poměr obvodů a velikost nejmenší opsaných pravoúhelníků. Li et al. (2004) pro hledání clusterů vychází z kombinace teorie grafů, Delaunayho triangulace, Voronoi diagramů a zákonů Gestaltismu.

Metodu DBSCAN, která nevyužívá grafů pro nalezení clusterů v prostorových datech představili Ester et al. (1996). Tato metoda byla původně určena pro shlukování bodových dat. Pro hledání shluků budov tuto metodu prakticky využili autoři Basaraner

a Selcuk (2008). Proces shlukování pomocí metody DBSCAN je řízen pouze pomocí dvou parametrů *d, ε*. Na počátku je vždy jedna z budov označena jako jádrová. Následně je hledán počet budov ve vzdálenosti *d* od jádrové budovy. Pokud počet budov do vzdálenosti *d* je větší nebo roven *ε*, tak jsou budovy nacházející se do vzdálenosti *d* od jádrové též označeny jako jádrové. Tento postup je rekurzivně volán na každou jádrovou budovu. V případě, že již bylo prozkoumáno okolí všech jádrových budov (ukončena rekurze), jsou jádrové budovy označeny jako jeden shluk. Následně je jako jádrová budova označena některá z dalších budov, která ještě nebyla zařazena do nějakého ze shluků. Komparativní studii algoritmů pro vyhledávání shluků představili Cetinkaya, Basaraner a Burghardt (2015).

Druhou skupinou algoritmů, které navazují na hledání shluků jsou algoritmy pro hledání vnitřní struktury shluků. Většina dosud popsaných metod se soustředila na vyhledávání předem známých a přesně specifikovaných uspořádání budov. Stávající studie ve většině případů využívají grafů pro určení sousedství budov stejně jako metody pro nalezení shluků. Pro určení vnitřního uspořádání poté využívají parametrů jako je např. blízkost a orientace. Christophe a Ruas (2002) se ve své práci soustřeďují na vyhledávání lineárních vzorů pomocí šablon. Zhang, Ai a Stoter (2012) ve své práci charakterizují vlastnosti pěti základních typů uspořádání (lineární, křivočarý, zarovnaný podél ulic, ve tvaru mřížky, neuspořádaný), avšak představují pouze algoritmus na rozeznání shluků podél ulic založený na Delaunayho triangulaci a minimální kostře grafů. Ve své další práci se Zhang et al. (2013) zaměřují na identifikaci lineárních a křivočarých typů uspořádání budov ve shlucích.

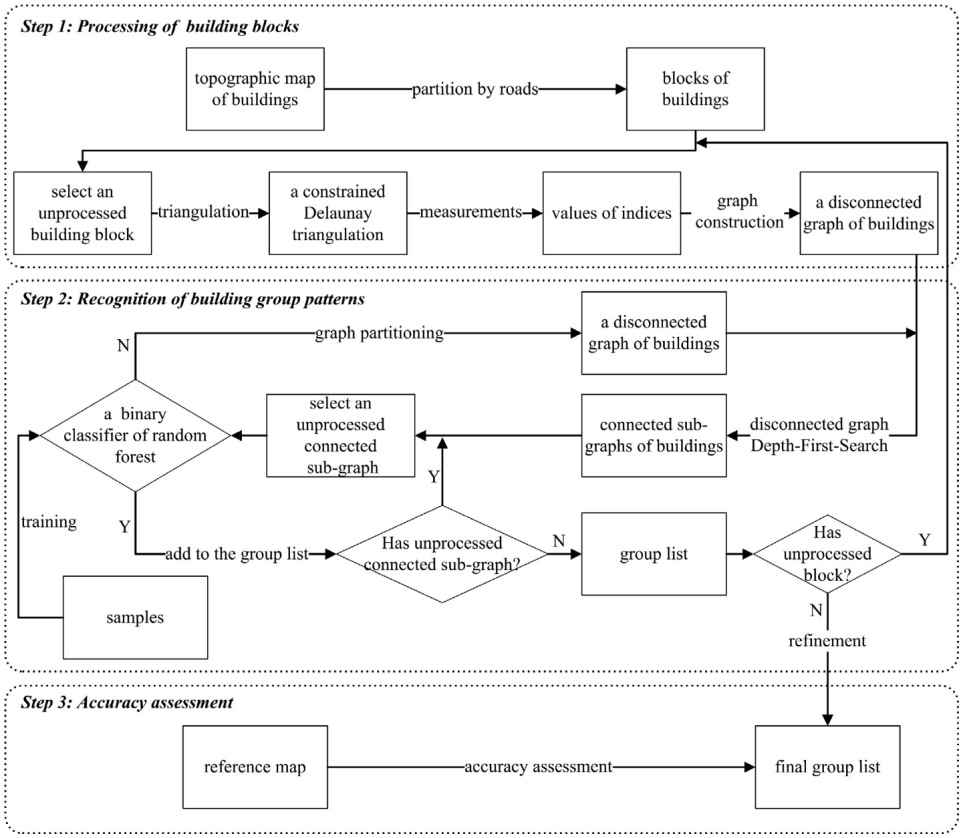
V historii vývoje automatické generalizace zástavby však bylo sepsáno velmi málo publikací, které by propojovaly metody na vyhledávání shluků a metody pro určení jejich vnitřního uspořádání. Vzhledem k tomu, že cílem této diplomové práce je vytvořit algoritmus, který bude propojovat obě části, bude několik dalších odstavců věnováno podrobnější rešerši několika vybraných článků, ve kterých se též autoři snaží o propojení procesu identifikace shluků zástavby a procesu klasifikace.



Obrázek 2: Schéma klasifikace typů zástavby
***Zdroj**:* Du et al., (2016)

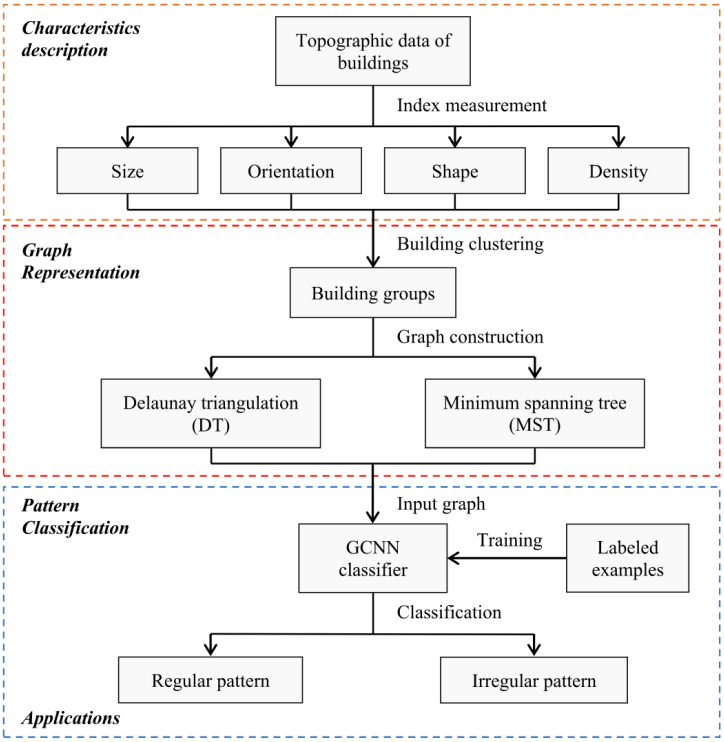
Autoři Du et al. (2016) ve své práci rozdělují proces identifikace shluků budov v zástavbě do čtyř kroků. V prvním kroku je Delaunayho triangulace použita pro zjištění sousedství. Pro triangulaci je využito zhuštěné pole lomových bodů polygonů budov. Ze vzniklých trojúhelníků jsou ponechány pouze ty, které spojují dvě rozdílné budovy. Hrany a vrcholy trojúhelníků jsou poté převedeny do podoby grafu. Ve druhém kroku jsou poté vypočteny podobnostní charakteristiky. Podobnostní charakteristiky představují atributy hran vytvořeného grafu. Do výpočtu podobnostních charakteristik vstupují zejména charakteristiky uzlových bodů, která daná hrana spojuje. Jinak řečeno, do výpočtu vstupují charakteristiky budov spojených danou hranou. Podobnostní charakteristiky představené v tomto článků jsou rozděleny na plošné charakteristiky, tvarové charakteristiky a vzdálenostní charakteristiky. Ve třetím kroku je využit algoritmus Relief-F (Kononenko 1994) pro určení váhy (důležitosti) každé podobnostní charakteristiky pro vytvoření shluků budov. Pro zjištění důležitosti jednotlivých atributů, využili autoři ručně segmentovaných shluků budov. Při znalosti shluků budov lze hrany grafu rozdělit na ty, které spojují budovy náležící stejnému shluku budov a na hrany spojující budovy odlišných shluků budov. Algoritmus Relief-F poté přiřazuje vyšší hodnotu těm podobnostním charakteristikám, u kterých rozdílné hodnoty dané podobnostní charakteristiky lépe poukazují na fakt, že se dané dvě budovy nacházejí v odlišných shlucích budov. Posledním krokem procesu segmentace je využití víceúrovňového dělení grafů pro rozdělení budov do jednotlivých shluků (Karypis, Kumar 1998). Pro určení struktur shluků postupují autoři tzv. shora dolů, kdy využívají již vytvořených grafů a dalších čtyř charakteristik (podobnost, vzdálenost, kontinuita, rovnoběžnosti) pro systematické určení jednotlivých typů struktur, tak jak jsou zobrazeny na obrázku č. 2. Dle autorů hlavní přínos této metody tkví ve třech aspektech.

Prvním z nich je snaha identifikovat více typů vnitřního uspořádání shluků. Druhým aspektem jsou čtyři navržená podobnostní kritéria integrovaná do víceúrovňového dělení grafů pro co nejlepší určení shluků. Za třetí je to navržení čtyř strategií pro systematickou identifikaci jednotlivých vzorů shluků.



Obrázek 3: Schéma algoritmu pro segmentaci a klasifikaci zástavby
***Zdroj**: He, Zhang a Xin (2018)*

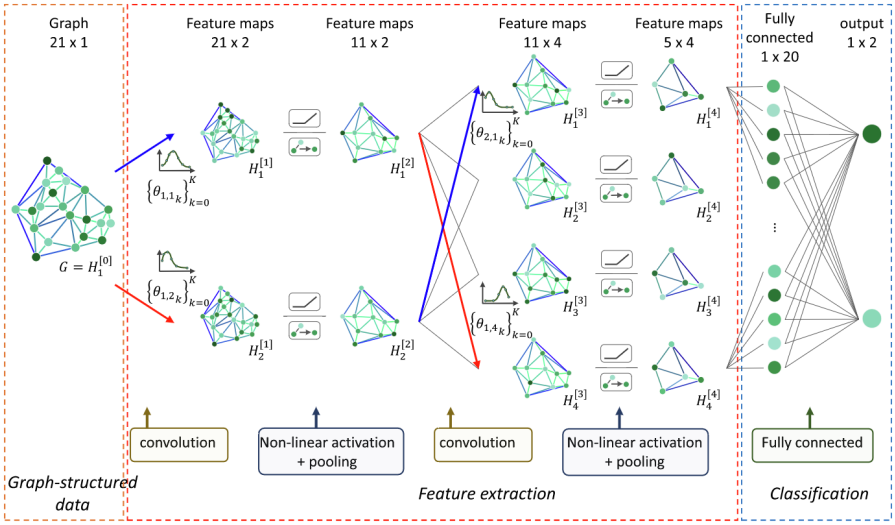
Autoři He, Zhang a Xin (2018) ve své práci představují metodu, jejímž cílem je identifikace jak pravidelných, tak nepravidelných typů zástavby. V prvním kroku jsou vstupní data (budovy) rozdělena do bloků pomocí silniční sítě. Na polygony budov v jednotlivých blocích je aplikována Delaunayho triangulace. Hrany vzniklé trojúhelníkové sítě, které překračují určitou délku nebo nesplňují některá další kritéria jsou odstraněny a zbylé jsou převedeny do podoby nespojitých grafů. Pro určení typu struktury shluku budov je v článku použita metoda náhodného lesa (*Random Forest*). Jako vstup do klasifikační metody použili autoři celkem pět charakteristik daných grafů (průměrná vzdálenost, směrodatná odchylka, poměr plochy budov k ploše konvexní obálky daných budov, poměr mezer mezi budovami k plochám budov, poměr plochy budov k ploše minimálního opsaného pravoúhelníku). Pokud není graf pomocí metody náhodného lesa zařazen do žádné kategorie je následně rozdělen na podgrafy, které opět vstupují do klasifikačního algoritmu. Schéma celé metody je znázorněno na obrázku č. 3.



Obrázek 4: Schéma metody pro segmentaci a klasifikaci zástavby
***Zdroj**:* Yan et al., (2019)

Cílem autorů Yan et al. (2019) je představit metodu pro klasifikaci zástavby založenou na neuronových sítích. Pro segmentaci budov do shluků se autoři rozhodli využít silniční síť, a to zejména z důvodu urychlení a zjednodušení procesu segmentace zástavby. Vlastní klasifikaci lze poté rozdělit do tří kroků, které jsou znázorněny na obrázku č. 4. V prvním kroku je pro každou budovu v daném shluku spočítáno celkem 11 charakteristik náležících do skupin – velikost, orientace, tvar a hustota. Následně jsou centroidy budov spojeny do podoby grafů. Pro tvorbu grafů autoři využívají dvou principů, a to Delaunayovy triangulace a minimální kostry grafů. Každý centroid budovy představuje uzel vzniklého grafu. Každému uzlu je poté přiřazeno 11 spočtených atributů. Hlavní přínos autorů spočívá ve využití GCNN (*Graph convolutional neural network*). GCNN jsou speciální kategorií neuronových sítí určených pro analýzu grafů pomocí Fourierových transformací a konvoluce. Schéma použité GCNN lze vidět na obrázku č. 5. Z obrázku lze vyčíst, že autoři využili dvou konvolučních vrstev následovaných aktivační vrstvou a tzv. pooling layer (viz kapitola 2.4). Autoři se ve svém výzkumu rozhodli využít pouze binární klasifikaci zástavby na pravidelné a nepravidelné vzory. Proto má schéma GCNN na svém konci pouze dva neurony. Jako testovací vzorek budov posloužil autorům dataset z čínského města Guangzhou. Výsledky práce autoři zároveň porovnali s dvěma dalšími metodami založenými na strojovém učení (SVM, RF),

které byly popsány autory Liqiang et al., (2013) a He, Zhang a Xin (2018). Výsledky ukazují vysokou úspěšnost všech tří metod při použití v oblasti, kde byly jednotlivé metody natrénovány. Avšak při aplikaci na jiná data prokazuje metoda založená na GCNN lepší schopnost generalizace a přizpůsobení se a tím i lepších výsledků než dvě zbývající. V diskusi autoři rozebírají tři možné přístupy ke klasifikaci zástavby. První přístup je založený na manuálním stanovení přesných pravidel pro klasifikaci shluků do jednotlivých typů. Právě stanovení přesných pravidel je časově velmi náročné a zároveň velmi složité pro použití v geograficky rozdílných oblastech. Druhou skupinou jsou metody založené na strojovém učení. Tyto metody vymezují hranice pro zařazení jednotlivých shluků automaticky a tím urychlují vlastní klasifikaci. Dle autorů jsou však tyto metody velmi citlivé na vzorová data a jejich množství. Poslední skupinou jsou metody založené na GCNN. Dle autorů tyto metody ještě více urychlují a zjednodušují proces klasifikace i z důvodu, že využívají statistiky o jednotlivých budovách, a ne celých shlucích jako předcházející metody. V závěru poté autoři představují možnosti pro vylepšení metody týkající se zejména využití efektivnějšího návrhu sítě a rozšíření klasifikovaných typů.



Obrázek 5: Schéma neuronové sítě pro identifikaci typu zástavby
***Zdroj**:* Yan et al., (2019)

Obdobně jako u předchozích článků je cílem autorů Hecht et al. (2013) klasifikovat zástavbu, avšak na rozdíl od již zmíněných článků je cílem autorů klasifikovat zástavbu více z urbanistického pohledu, než s důrazem na kartografickou generalizaci zástavby. Celkem autoři vymezili 8 typů zástavby jako např. terasové domy, řadové domy a industriální budovy. Navržená metoda se od předchozích liší též vynecháním procesu segmentace. Do procesu klasifikace tak nevstupují shluky budov, ale pouze jednotlivé

budovy. Pro klasifikaci budov představili autoři řadu popisných charakteristik, které se vždy vztahují ke konkrétní budově. Pro vlastní klasifikaci využili autoři několik klasifikačních algoritmů spadajících do oblasti strojového učení. Konkrétně autoři porovnali algoritmy KNN (*K-nearest neighbor classifier*), CART (*Classification and Regression Tree*), BAGGING *(Bagging Trees*), RF (*Random Forest algorithm*) a SVM (*Support Vector Machine)*. Po klasifikaci jsou jednotlivé budovy agregovány do bloků pomocí dostupné vrstvy městských bloků z databáze ATKIS. Každému bloku je poté přiřazen typ zástavby podle převažujícího typu zástavby jednotlivých budov náležících do daného bloku. Autoři ve své práci porovnávali jednotlivé klasifikátory jak na úrovni jednotlivých budov, tak po agregaci na úroveň bloků. V obou případech nejvyšší přesnosti dosáhl klasifikátor RF. Klasifikační přesnost dosáhla v obou případech hodnoty přibližně 80 %. V závěru autoři poukazují na možné využití představené metody v oblasti územního plánování. Zároveň však poukazují na nepříliš vysokou klasifikační přesnost některých typů zástavby.

2.3 Strojové učení

Strojové učení (*machine learning*) je vědní obor spadající pod oblast umělé inteligence (*AI*). Hlavním předmětem studia tohoto oboru je tvorba statistických modelů, které počítačové systémy využívají pro provedení specifických úkolů bez jasně stanovených instrukcí a pravidel (Bishop 2006). Běžné matematické modely mají známé vstupy a rozhodovací pravidla a na základě toho hledají odpovědi. Algoritmy strojového učení naopak pomocí známých vstupů a odpovědí hledají (učí se) rozhodovací pravidla (Goldberg, Holland 1988), která následně aplikují na další vstupní data.

Mezi základní techniky algoritmů strojového učení patří **učení s učitelem** (*supervised learning*) a **učení bez učitele** (*unsupervised learning*). Cílem učení s učitelem je odhalení vztahů mezi vstupními daty a známými výstupy. Podle typu výstupu lze dělit učení s učitelem na klasifikaci nebo regresi. Při klasifikaci je každému vstupnímu prvku přiřazena třída. Při regresi je na výstupu číslo. Příkladem regrese může být určení dojezdu automobilu v závislosti na množství paliva, spotřebě a dalších vstupních proměnných. Mezi algoritmy využívající učení s učitelem patří *Support vector machines (SVM)* (Cortes, Vapnik 1995)*, Random Forest (RF)* (Breiman 2001) a logistická regrese (LR) (Menard 2002).

V případě učení bez učitele nejsou pro vstupní data známy výstupy. Algoritmy využívající učení bez učitele mají nejčastěji za úkol nalézt struktury ve vstupních datech. Příkladem problému řešeného pomocí učení bez učitele je shluková analýza. Cílem této analýzy je roztřídit vstupní data do tříd, avšak na rozdíl od klasifikace v případě učení s učitelem, nejsou tyto třídy předem známy. Příkladem algoritmu pro shlukovou analýzu je *K-means* (Mannor et al. 2011). Kromě shlukových analýz lze využít algoritmy využívající učení bez učitele např. pro redukci počtu dimenzí dat. Příkladem takového algoritmu je analýza hlavních komponent (*PCA*) (Maćkiewicz, Ratajczak 1993).

Vzhledem k tomu, že z algoritmů strojového učení je dále v práci využita logistická regrese, je další kapitola věnována bližšímu popisu jejího fungování. Důvody pro zvolení logistické regrese jsou popsány v kapitole 4.7.

2.3.1 Logistická regrese

Logistická regrese je jeden z matematických modelů sloužící pro účely klasifikace. Na rozdíl od lineární regrese, kdy je na základě hodnot nezávislých proměnných vypočítávána hodnota spojité závislé proměnné, slouží logistická regrese pro modelování pravděpodobnosti jevu kategorické závislé proměnné na základě hodnot nezávislých proměnných (Řeháková 2000).

Původně byla logistická regrese navržena pro modelování pravděpodobnosti jevu binární závislé proměnné (*binární logistická regrese*). Příkladem může být odhadování vzniku srdeční choroby na základě parametrů jako je věk, BMI, krevní tlak atd. Logistickou regresi lze však též využít pro predikci více než dvou tříd (*multinomická logistická regrese*) a také závislé ordinálního typu (*ordinální logistická regrese*). Pro jednodušší interpretaci principů logistické regrese, je její fungování popsáno na příkladu binární logistické regrese. To znamená, že nezávislá proměnná *Y* nabývá pouze dvou hodnot. V případě, že daný jev nastane, tak *Y(x) = 1,* v opačném případě *Y(x) = 0*.

Pro modelování vztahu mezi nezávislými proměnnými a závislou proměnnou využívá logistická regrese tzv. logistickou funkci. Tuto funkci lze zapsat ve tvaru:

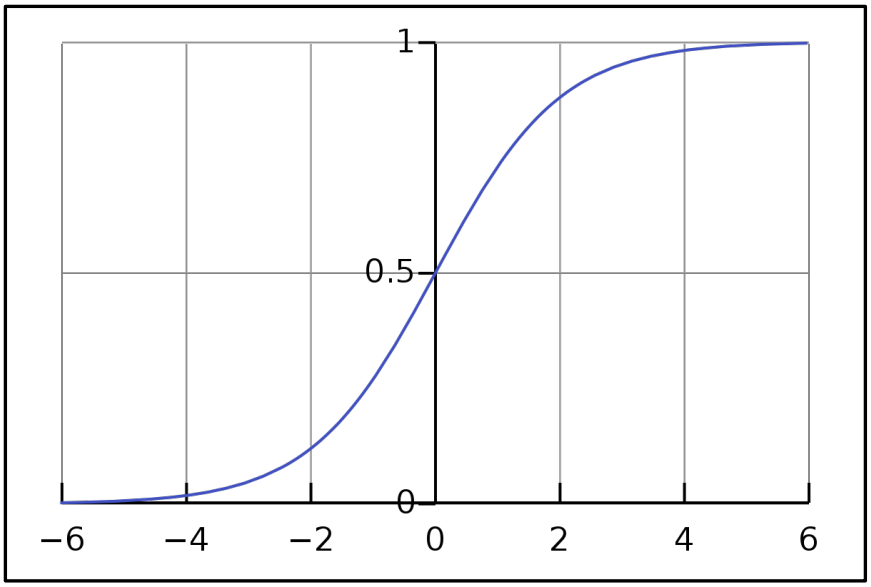
$$\ln\left(\frac{p(x)}{1-p(x)}\right) = \beta' x,$$

kde x jsou hodnoty nezávislé proměnné. Vektor β je poté vektorem neznámých parametrů. Hodnota $p(x)$ představuje pravděpodobnost, že *Y(x) = 1*. Hodnota $1-p(x)$ poté představuje opačný případ, tj. pravděpodobnost jevu *Y(x) = 0.* Celý zlomek na levé straně poté představuje šanci (*odds*), neboli poměr pravděpodobnosti, kdy daný jev

nastane oproti pravděpodobnosti, že daný jev nenastane. Z definice šance vychází, že je vždy větší než 0. Logistická regrese však vyjadřuje pravděpodobnost. Z tohoto důvodu je využito přirozeného logaritmu, který omezí rozsah, jakého může levý člen nabývat, na interval (0;1) (*log-odds*). Celou rovnici lze poté přepsat do tvaru:

$$p(x) = \frac{e^{\beta' x}}{1 + e^{\beta' x}}.$$

Takto zapsaná logistická funkce poté představuje rovnici logistické křivky neboli křivky *sigmoid* (obrázek 6). Přesný tvar křivky je poté určen pomocí regresních koeficientů β. Jejich počet je závislý na počtu vstupních proměnných. Narozdíl od lineární regrese, kdy je pro stanovení regresních koeficientů využito metody nejmenších čtverců, je v tomto případě využito metody maximální pravděpodobnosti. Pro konečné určení koeficientů se využívají iterační metody, jako je například Newtonova metoda. Proces výpočtu koeficientů bývá nazýván jako fitování (Menard 2002).



Obrázek 6: Logistická křivka
***Zdroj**: Wikipedia (2020)*

2.4 Neuronové sítě

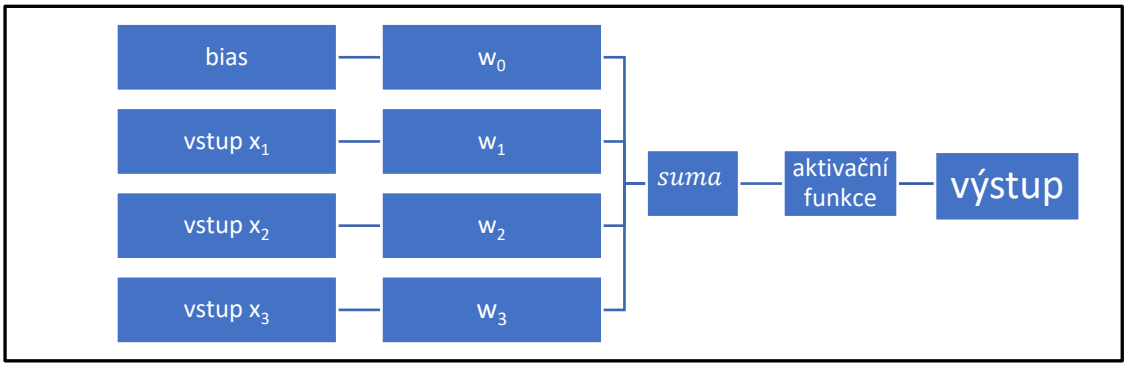
Umělé neuronové sítě (*Artificial Neural Networks*) jsou specifickou podoblastí strojového učení (*Machine Learning*). Umělá neuronová síť je model založený shlukování propojených uzlů – umělých neuronů, který je inspirován fungováním lidského mozku. Spojení mezi umělými neurony, stejně jako synapse v lidském mozku, dokáže přenášet signály neboli informace. Umělé neurony signály zpracovávají a posílají ho dalším propojeným neuronům (Deep AI 2018).

2.4.1 Model neuronu

Jeden z prvních modelů neuronu byl navržen již ve 40. letech 20. století, avšak až rychlý vývoj techniky v posledních desetiletích umožnil smysluplné využití neuronových sítí (Tišnovský 2018). Každý neuron (obrázek 7) může mít na vstupu libovolný počet vstupů a pouze jeden výstup. Výpočet uvnitř neuronu lze matematicky zapsat jako:

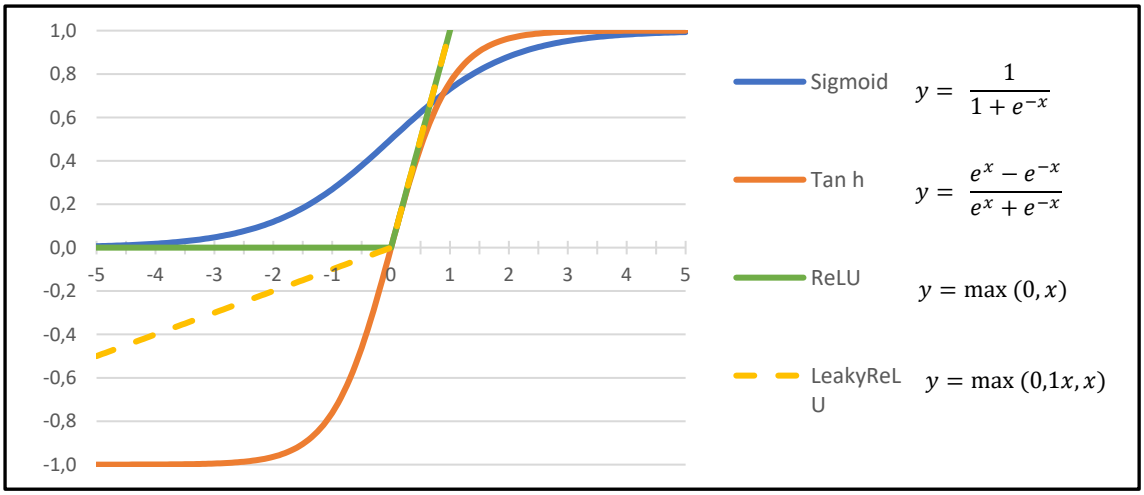
$$f(x, w) = \phi\left(\sum w_i * x_i\right),$$

kde x je vstupní hodnota, w představuje váhu daného vstupu a ϕ je **aktivační funkce**. Aktivační funkce převádí vstupní hodnotu nejčastěji do intervalu <-1;1> nebo <0;1>, čímž odstraňuje lineární vztah mezi vstupními hodnotami a výsledkem. V současnosti je nejvíce využívanou aktivační funkcí skrytých neuronů ReLU (*Rectified Linear Unit*), jejíž graf lze společně s dalšími používanými aktivačními funkcemi vidět na obrázku č. 8. (Ramachandran, Zoph, Le 2017).



Obrázek 7: Schéma umělého neuronu
***Zdroj**: vlastní zpracování*

Pro lepší schopnost generalizace neuronové sítě se do neuronu připojuje takzvaný **bias**. Bias je ve své podstatě další vstup (obrázek 7), jehož hodnota je vždy rovna jedné a mění se pouze váha. Díku tomu dokáže bias „posouvat" křivku aktivační funkce napravo či nalevo v závislosti na váze (Heaton 2019).

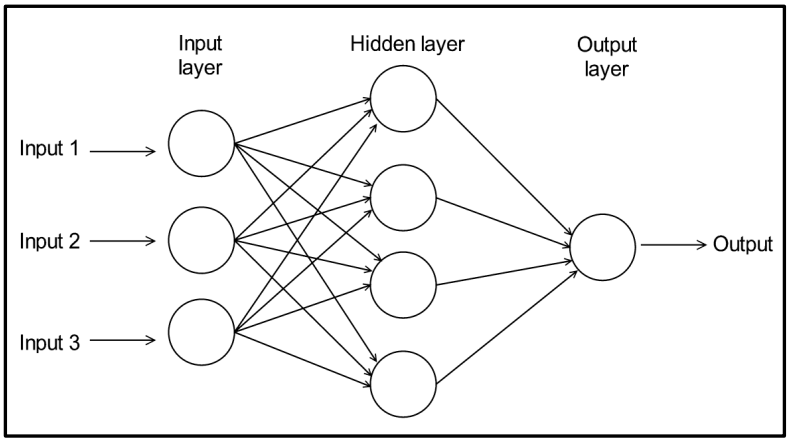


Obrázek 8: Aktivační funkce
***Zdroj**: vlastní zpracování*

2.4.2 Obecné schéma neuronové sítě

Jednotlivé neurony jsou uvnitř neuronové sítě skládány do vrstev. Každý neuron z jedné vrstvy je vždy spojen se všemi neurony ve vrstvě následující. Síla každého spojení představuje váhu, kterou jsou upravovány vstupy do neuronu, tak jak to bylo popsáno v předchozí podkapitole. Podle umístění vrstvy uvnitř sítě lze rozlišovat tři základní typy vrstev (obrázek 9)(Tišnovský 2018).

První vrstvou je tzv. **vstupní** (*input layer*). Cílem této vrstvy je načtení vstupních dat a jejich předání první skryté vrstvě. Neurony ve vstupní vrstvě vstup nijak neupravují, neboť neobsahují aktivační funkci, ani bias. Počet neuronů ve vstupní vrstvě vždy odpovídá počtu proměnných vstupních dat. Vzhledem k tomu, že tato vrstva pouze předává vstup první skryté vrstvě, bývá často při popisu konkrétních architektur neuronových sítí vynechávána a jako první vrstva neuronové sítě je uváděna první skrytá vrstva.



Obrázek 9: Základní typy vrstev v neuronové síti
***Zdroj**: Do, Taherifar, Vu (2019)*

Další vrstvou v neuronové sítě je již zmíněná **skrytá vrstva** (*hidden layer*). Neurony ve skryté vrstvě již přesně odpovídají fungování neuronu popsaného v předchozí podkapitole. Počet skrytých vrstev v síti není nijak omezen. Jejich počet se pohybuje od jednotek až po stovky v případě hlubokých neuronových sítí (*deep neural networks*).

Poslední vrstvou je tzv. **výstupní** (*output layer*). Stejně jako v případě skrytých vrstev, jsou i neurony ve výstupní vrstvě napojeny na všechny neurony v předcházející vrstvě. Neurony v poslední vrstvě se liší použitou aktivační funkcí, která je stejně jako jejich počet, závislá na účelu, ke kterému neuronová síť slouží. V případě, že je použita pro regresi, tvoří poslední vrstvu pouze jeden neuron a jeho aktivační funkce je *lineární* ($y=x$). V případě klasifikace odpovídá počet neuronů ve výstupní vrstvě počtu klasifikovaných tříd (n). Aktivační funkcí je v tomto případě *Softmax*. Tato aktivační funkce vypočítává pravděpodobnost zařazení daného prvku do konkrétní třídy pomocí vzorce:

$$y_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}},$$

kde x_i je suma hodnot vstupujících do i-tého neuronu výstupní vrstvy včetně biasu. Index n poté označuje počet neuronů ve výstupní vrstvě. Z rovnice též vyplývá, že součet pravděpodobností zařazení daného prvku do jednotlivých tříd je vždy roven jedné.

2.4.3 Učení neuronové sítě

Při učení neuronových sítí lze, stejně jako u jiných algoritmů strojového učení, rozlišovat dva základní postupy. Prvním postupem je tzv. učení bez učitele. Tyto neuronové sítě mohou být použity například pro hledání shluků ve vstupních datech. Příkladem tohoto typu jsou tzv. *samoorganizující se mapy*, které se podle autora označují jako *Kohonenovy mapy* (Kohonen 1982). Druhým postupem je tzv. učení s učitelem. Základním typem neuronových sítí využívajících učení s učitelem jsou **dopředné neuronové sítě** *(Feed forward Neural networks)*(Hornik, Stinchcombe, White 1989). Tento typ sítí se využívá jak pro regresní, tak pro klasifikační účely. Mezi další typy neuronových sítí patří **rekurentní neuronové sítě** *(Recurent Neural networks)*, mezi které se řadí LSTM sítě (*Long short-term memory*). Tento typ sítí se používá například pro rozpoznávání řeči (Sutskever, Vinyals, Le 2014). Vzhledem k cílům této diplomové práce budou následující odstavce a kapitoly věnovány dopředným neuronovým sítím se zaměřením na klasifikaci.

Při učení neuronové sítě s učitelem lze rozeznávat dva základní procesy. Prvním procesem je **dopředná propagace** (*forward propagation*). Při tomto procesu dochází k načtení dat první vrstvou a jejich průchodu celou neuronovou sítí až k závěrečné vrstvě. Při prvním průchodu dat neuronovou sítí je síla jednotlivých spojení mezi neurony zvolena náhodně. Výstup z poslední vrstvy bývá označován jako predikce. Pro určení, s jakou chybou neuronová síť predikovala je predikce porovnána se správným výsledkem (tabulka 1). Pro určení velikosti chyby slouží **ztrátová funkce** (*loss function*). Pro klasifikaci více než dvou typů používá **kategoriální křížová entropie** (*categorical cross entropy*), kterou lze matematicky zapsat jako:

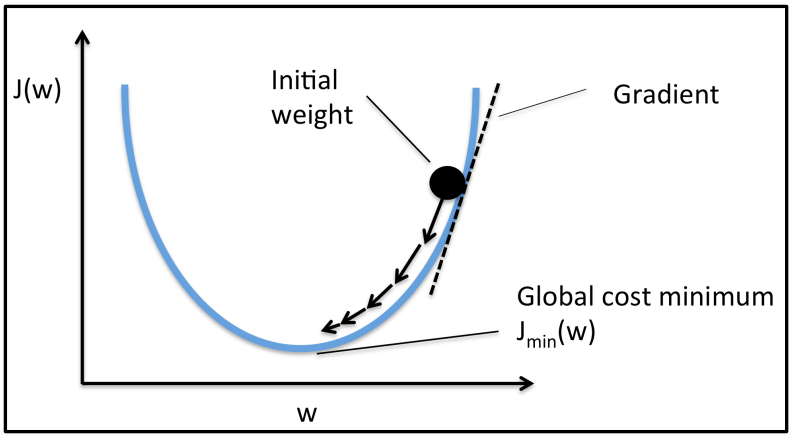
$$ztráta = -\frac{1}{N} * \sum_{i=1}^{N} y_i * \ln(\hat{y}_i)\,,$$

kde y_i je predikovaná pravděpodobnost, $\hat{y}_i$ je správný výsledek a N je počet klasifikovaných tříd.

	Predikce	**Správný výsledek**
Třida 1	0,3	1,0
Třída 2	0,1	0,0
Třída 3	0,6	0,0

Tabulka 1: Ztrátová funkce
***Zdroj**: vlastní zpracování*

Velikost této ztráty poté řídí druhý proces při učení sítě s učitelem, kterým je **zpětná propagace** (*backpropagation*). Cílem tohoto procesu je pomocí úpravy jednotlivých vah uvnitř sítě minimalizovat velikost hodnoty ztrátové funkce při dalším dopředném průchodu dat sítí. Pro minimalizaci ztrátové funkce se v případě neuronových sítí využívá algoritmus *gradient descent*. Jedná se o iterační algoritmus pro minimalizaci funkce založený na výpočtu derivací. Tento algoritmus je založený na hledání směru, ve kterém daná funkce klesá nejrychleji (obrázek 10)(Heaton 2019). Cílem algoritmu *gradient descent* je nalézt globální minimum funkce. Konkrétní vyhledávací technika pro určení směru poklesu funkce je určena pomocí **optimalizátoru** (*Optimizer*). Příkladem konkrétních optimalizátorů jsou: *SGD*, *RMSprop*, *Adam* (Kingma, Ba 2015).



Obrázek 10: Gradient descent
***Zdroj**: Raschka (2014)*

Proměnné v neuronové síti, které jsou neuronovou sítí samovolně upravovány, se nazývají **parametry** sítě. Proměnné, jejichž hodnotu je nutné před procesem učení stanovit, se poté nazývají **hyperparametry** sítě. Stejné označení se používá pro všechny algoritmy v oblasti strojového učení (Brownlee 2019). Příkladem hyperparametru je *Learning rate*. Tento hyperparametr ovlivňuje míru úpravy vah v sítí. Hodnota tohoto hyperparametru silně ovlivňuje rychlost učení. Vyšší hodnota *Learning rate* způsobuje větší míru úpravy vah a tím urychluje proces učení. Při příliš velké hodnotě *Learning rate* však nemusí algoritmus nalézt globální minimum. Velmi malé hodnoty *Learning rate* naopak velice zpomalují celý proces učení. Velmi malá hodnota *Learning rate* může též způsobit, že bude nalezeno pouze lokální minimum, ne však globální (Heaton 2019).

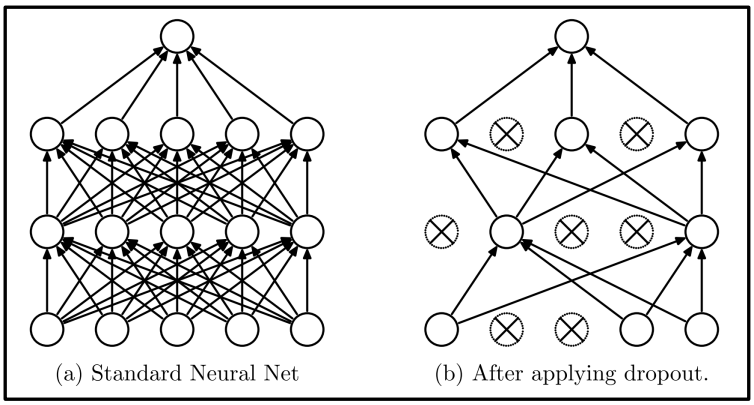
Počet dopředných propagací předtím, než jsou upraveny váhy uvnitř neuronové sítě je určen pomocí hyperparametru *Batch Size* (velikost dávky). Lze též říct, že tento hyperparametr určuje počet záznamů ze vstupních dat, které jsou současně využity pro jednu dopřednou propagaci. Při zpětné propagaci jsou poté váhy upravovány na základě ztrát všech záznamů v jedné dávce. Moment, kdy jsou pro naučení sítě využity všechny záznamy ve vstupních datech se nazývá **epocha**. Pro naučení neuronové sítě se obvykle využívají desítky až tisíce epoch.

2.4.4 Typy vrstev v neuronové síti

V kapitole 2.3.1 byly představeny typy vrstev podle jejich umístění v neuronové síti. Cílem této podkapitoly práce bude představit několik typů vrstev podle jejich vlastností a účelu v neuronové síti (Ng 2019), (Heaton 2019).

1. **Fully-conected/Dense layer**

Tato vrstva je složena z neuronů, tak jak byly popsány v kapitole 2.4.1. Název této vrstvy vyplývá z faktu, že každý z neuronů této vrstvy je navázán na každý neuron z předchozí vrstvy a zároveň je každý neuron této vrstvy navázán na každý neuron v další vrstvě (obrázek 11, vlevo). Jako *fully-connected* bývají též označovány úvodní i závěrečná vrstva.



Obrázek 11: Dropout
***Zdroj**: Srivastava et al. (2014)*

2. **Dropout layer**

Na rozdíl od předchozí vrstvy, se tento typ neskládá z neuronů. Tato vrstva též neobsahuje žádný parametr, který by byl během procesu učení upravován. Cílem této vrstvy v neuronové síti je zabránit **přeučení sítě** (*overfitting*), neboli stavu, kdy daná neuronová síť dokáže velmi dobře reagovat na známé prvky nacházející se v trénovací množině. Avšak v případě, že neuronová síť má predikovat výsledek záznamu, na kterém nebyla učena, je výsledek mnohem horší. Zejména hluboké neuronové sítě mají tendenci predikovat výsledek pouze na základě několika silných spojení. To během procesu učení vede k tomu, že váhy silných spojení jsou zpětnou propagací ještě více posilovány, kdežto zbylé jsou více utlumovány (Brownlee 2019). *Dropout* vrstva se snaží předcházet tomuto stavu vypínání určitých spojení během procesu učení (obrázek 11, vpravo). Z obrázku je též patrné, že *Dropout* není vrstva v pravém slova smyslu, její chování lze připodobnit k aktivační funkci. Vypnutí spojení poté znamená, že hodnota výstupu z neuronu je rovna nule. Počet vypnutých spojení (neuronů) je určen hyperparametrem této vrstvy, který je nutné stanovit před procesem učení. Výběr neuronů, které nebudou zapojeny do trénovaní neuronové sítě během konkrétní dávky je vždy náhodný. *Dropout* vrstva se používá jak ve spojení s *Fully-conected*, tak s dále představenou konvoluční vrstvou. *Dropout* je též možné aplikovat pouze na konkrétní vrstvu nebo na všechny vrstvy v neuronové síti.

Jedním z dalších způsobů, který se využívá pro předcházení přeučení sítě je tzv. *regularizace*. Obdobně jako *Dropout*, se i regularizace snaží odstranit riziko vzniku silných spojení. Regularizace se snaží tomuto efektu předcházet pomocí penalizace vysokých hodnot vah během procesu zpětné propagace. V oblasti neuronových sítí se nejčastěji využívá tzv. L2 (*ridge)* regularizace (Heaton 2019). Technika regularizace se však využívá i u jiných klasifikačních algoritmů jako je například již zmíněná logistická regrese. Obdobně jako u neuronových sítí je cílem lepší schopnost generalizace výsledného modelu. U logistické regrese regularizace upravuje, jakým způsobem jsou počítány regresní koeficienty (Lee et al. 2006).

3. **Batch normalization layer**

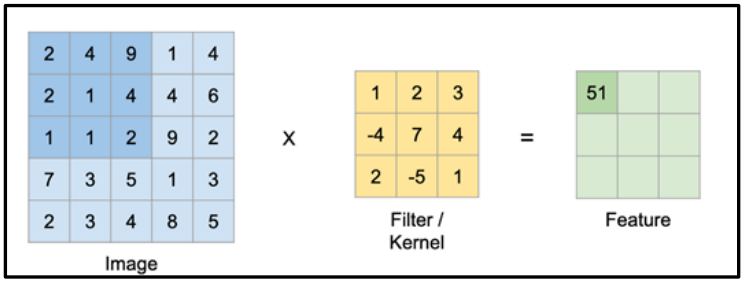
Jedním ze způsobů, jak urychlit proces učení sítě je využití **dávkové normalizace**. Cílem dávkové normalizace je odstranit tzv. *internal covariate shift* (Ioffe, Szegedy 2015). Obecným cílem normalizace dat je změna rozložení tak, aby průměr dat byl roven nule a směrodatná byla rovna jedné. Dávková normalizace spočívá v normalizaci výstupu z aktivační funkce. Výstup z aktivační funkce konkrétního neuronu je normalizován pomocí všech výstupů z této aktivační funkce pro jednu dávku dat. Obdobně jako v případě *Dropout* se nejedná o vrstvu v pravém slova smyslu, ale o další metodu, kterou je upravována hodnota na výstupu z neuronu. Dávkovou normalizaci je možné opět využít jak ve spojení s *Fully-conected*, tak konvoluční vrstvou. Dávkovou normalizaci je též možné použít pouze ve spojení s konkrétní vrstvou nebo pro všechny vrstvy v neuronové síti. Pro vyšší výkonost neuronové sítě využívá dávková normalizace ještě dvou parametrů, jejichž hodnota je upravována pomocí zpětné propagace. Konkrétně se jedná o *měřítko* γ (*scale*) a *posun* β (*shift*). Celý proces dávkové normalizace lze poté matematicky vyjádřit jako:

$$y_i = \left(\frac{x_i - \bar{x}_B}{\sqrt{\sigma_B^2}} \right) * \gamma + \beta.$$

4. **Convolution layer**

Vrstvou neuronových sítí specificky určenou pro zpracování obrazu je **konvoluční vrstva**. Tato vrstva byla představena poprvé autory (Lecun, Bengio 1998). Oproti *Fully-conected* vrstvě, která zpracovává 1D data, byla tato vrstva specificky navržena pro zpracování 2D dat (obrazu). Primárním účelem této vrstvy je detekovat prvky v obraze jako jsou hrany, linie a další vzory. Pro jejich detekci je v konvoluční vrstvě využito trénovatelných filtrů a principu konvoluce.

Filtr (*kernel*) je nejčastěji představován maticí hodnot o rozměrech *m*n*. Takto definovaný filtr provádí konvoluci přes celou výšku a šířku vstupního obrazu. Při každé změně polohy filtru je proveden skalární součin hodnot filtru a hodnot ve vstupním obraze odpovídajících aktuální poloze filtru. Výstupem skalárního součtu je vždy pouze jedna hodnota. Tato hodnota je poté upravena pomocí aktivační funkce. Upravené hodnoty jsou poté skládány do **výstupní vrstvy** (*feature map*) (obrázek 12). Obdobného principu se využívá v DPZ např. pro detekci hran v obraze.



Obrázek 12: Konvoluční vrstva
***Zdroj**: Krut (2019)*

V případě, že vstupní obraz obsahuje více pásem, probíhá konvoluce přes všechna pásma najednou. Z každé polohy filtru je tedy výstupem vždy pouze jedna hodnota. V jedné konvoluční vrstvě však může být více filtrů. Výstupní obraz z konvoluční vrstvy má poté tolik pásem, kolik filtrů obsahuje daná konvoluční vrstva.

To, jak se filtr přes vstupní obraz pohybuje definuje parametr *Stride*. Nejčastěji se filtr posunuje o jeden „pixel“. Je však možný i posun o více pixelů, avšak posun musí být vždy menší nebo roven rozměrům filtru. Parametr *Padding* poté charakterizuje chování filtru v krajních pozicích, respektive úpravu vstupního obrazu. Jedním ze způsobů zpracování vstupního obrazu je přidání nulových hodnot okolo vstupního rastru. To při hodnotě *Stride = 1* vede k zachování rozměrů vstupního obrazu. Druhou možností je nevyužití krajních hodnot. To má za následek částečnou ztrátu informace z původního obrazu a zmenšení rozlišení výstupního obrazu.

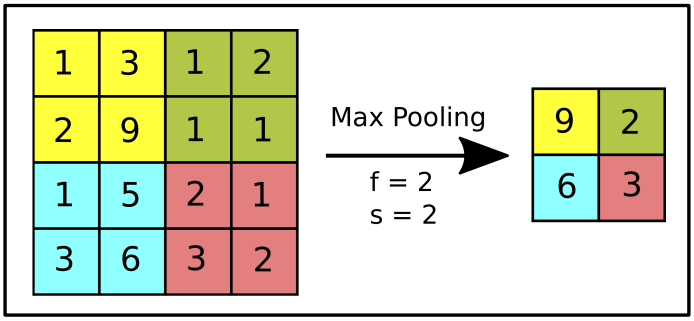
Jak již bylo řečeno, hodnoty koeficientů (parametry) jednotlivých filtrů jsou trénovatelné. To znamená, že jejich hodnoty jsou při zpětné propagaci upravovány stejně jako váhy spojení ve *Fully-conected* vrstvě. Celkový počet trénovatelných parametrů jedné vrstvy lze vypočíst jako:

$$počet\ trénovatelných\ parametrů = (m * n * h + 1) * f,$$

kde *m, n* jsou rozměry filtru. Proměnná *h* označuje počet pásem vstupního obrazu konvoluční vrstvy a proměnná *f* poté značí počet filtrů dané konvoluční vrstvy. Jednička je ve vzorci je kvůli přítomnosti biasu.

5. **Max Pooling layer**

Cílem této vrstvy je redukovat rozlišení výstupních obrazů z konvolučních vrstev. Pomocí redukce rozlišení dochází zejména k poklesu vah (parametrů sítě), která musí neuronová síť upravit. To vede zejména ke zrychlení procesu učení. Cílem této vrstvy je též předcházet přeučení sítě (Deep AI 2018).



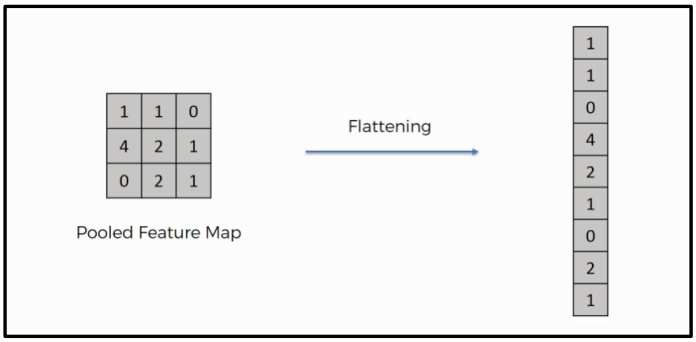
Obrázek 13: Max Pooling layer
***Zdroj**:* Deshpande (2016)

Obdobně jako u konvoluční vrstvy využívá i tato vrstva pohybujícího se okna. V tomto případě však nejsou hodnoty ve vstupním obraze, které odpovídají aktuální pozici pohybujícího se okna, upravovány pomocí hodnot filtru, ale je z nich vybrána pouze maximální hodnota. Plovoucí okno u tohoto typu vrstvy zpracovává vždy pouze jedno pásmo vstupního obrazu najednou. Z tohoto důvodu je počet pásem ve výstupním obraze z této vrstvy roven počtu pásem vstupního obrazu. U tohoto typu vrstev se též nenachází aktivační vrstva ani bias.

Posun pohybujícího se okna je opět definován pomocí parametru *Stride*. V případě tohoto typu vrstev se většina hodnota *Stride* rovná rozměrům pohybujícího se okna. (obrázek 13). Chování pohybujícího se okna na okrajích obrazu, stejně jako v případě konvoluční vrstvy, řídí parametr *Padding.*

6. **Flattening layer**

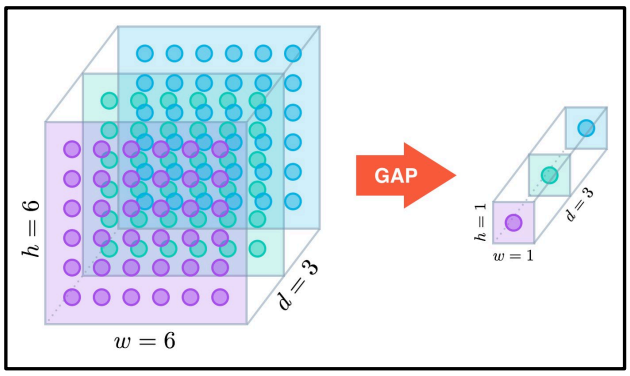
Cílem této vrstvy je převod výstupu z konvoluční, respektive *Max Pooling* vrstvy do podoby, kterou může zpracovat *Fully-conected* vrstva. Neboli jedná se o převod třírozměrné matice pouze do jednorozměrné (obrázek 14). Na obrázku č. 14 je pro názornost převod pouze dvourozměrné matice, avšak lze předpokládat, že poslední konvoluční vrstva bude mít více než jeden filtr, a tudíž bude obraz na výstupu vícepásmový. Při tomto převodu jsou zachovány všechny hodnoty, které do této vrstvy v podobě vícerozměrné matice vstupují.



Obrázek 14: Flattening layer
***Zdroj**:* Escontrela (2017)

7. Global average pooling layer – GAP

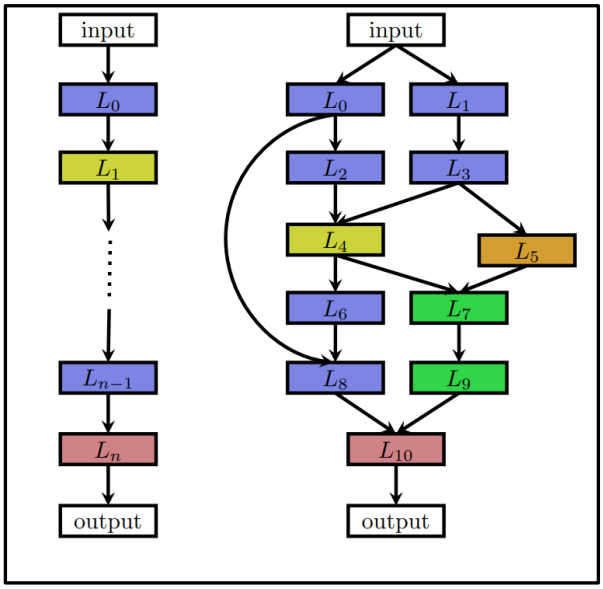
Tato vrstva plní stejný účel jako předchozí zmíněný typ vrstvy. Na rozdíl od něj však nezachovává všechny hodnoty, které v podobě vícerozměrné matice do této vrstvy vstupují. Po aplikaci několika *Max Pooling* vrstev v síti mají obvykle výsledné obrazy již relativně malé rozlišení, avšak většinou obsahují velký počet pásem (obrázek 17). Z tohoto důvodu je ze vstupního vícepásmového obrazu zachována pouze průměrná hodnota každého pásma (obrázek 15). Tento způsob oproti *Flattening* vrstvě vede ke snížení prvků ve výsledné jednorozměrné matici. Snížení množství prvků má podobné cíle jako *Max Pooling* vrstva. Nižší počet prvků sice znamená částečnou ztrátu informace, avšak vede k rychlejšímu procesu učení a předchází přeučení sítě (Cook 2017).



Obrázek 15: Global average pooling
***Zdroj**:* Cook (2017)

2.4.5 Architektura neuronové sítě

Architektura sítě určuje, ze kterých typů vrstev se skládá a jaké je jejich uspořádání v rámci sítě. Hyperparametry jsou poté proměnné jednotlivých vrstev a některé další parametry sítě, jako je *Learning Rate.* Hledání ideální architektury sítě je v literatuře označována jako *NAS* (*neural architecture search*). Hledání ideální architektury sítě je silně provázáno s hledáním hyperparametrů sítě, neboť různé architektury vyžadují jiné hodnoty hyperparametrů. Proto jsou tyto dva procesy velmi často spojovány do jediného (Long, Zhang, Zhang 2019). Wistuba, Rawat a Pedapati (2019) ve své práci definují tři fáze tohoto procesu. První fáze je definice prohledávaného prostoru. Cílem této fáze je definovat, jaké typy vrstev mohou být v sítí použity, jak mohou být jednotlivé vrstvy propojeny a jakých hodnot mohou nabývat jejich proměnné (hyperparametry). Druhou fází tohoto procesu je volba vyhledávací strategie. Konkrétní strategie pro hledání optimálních hyperparametrů jsou představeny v kapitole 2.6. Posledním krokem je stanovení metriky, podle které bude posuzována vhodnost konkrétní architektury. Kromě základních metrik jako je přesnost klasifikace, může být důležitý též čas nebo výkonnostní náročnost, ve smyslu systémových prostředků počítače.



Obrázek 16: Architektura sítě
***Zdroj**:* Zoph et al. (2018)

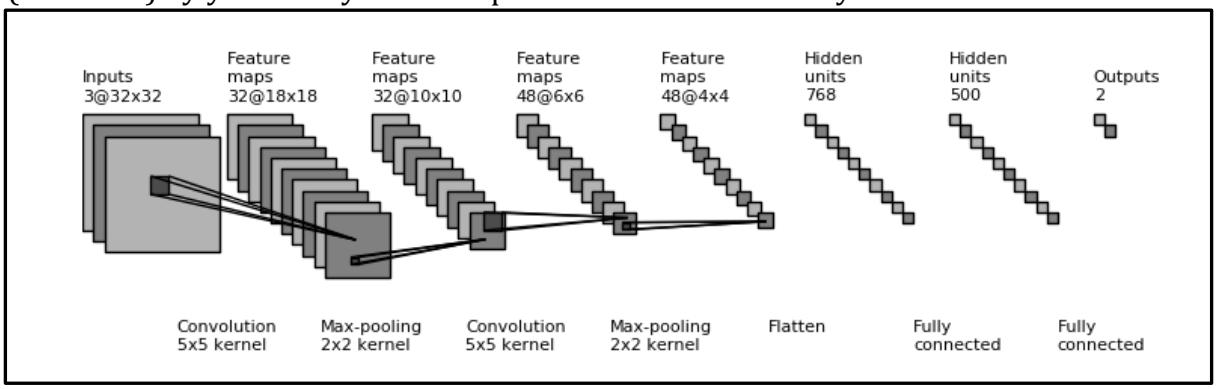
Mezi nejjednodušší architektury sítě patří takzvané řetězení vrstev (obrázek 16, vlevo). Při tomto způsobu je vstup do vrstvy L_i představován výstupem vrstvy L_{i-1}. Prohledávací prostor v tomto případě definuje, jakého typy jednotlivé vrstvy mohou být (*Dense, Dropout, Convolution, …*), viz kapitola 2.4.4 a jakých hodnot mohou nabývat jejich hyperparametry. V současné době dochází k rozvoji architektur sítí, které využívají

mnohem složitější napojení jednotlivých vrstev (obrázek 16, vpravo). Tyto architektury jsou využívány zejména v oblasti rozpoznávání obrazu. V tomto případě je každá vrstva představována funkcí $g_i(L_{i-1}^{out}, \dots L_0^{out})$, která může kombinovat výstupy více předcházejících funkcí (vrstev). Definování vyhledávacího prostoru je v tomto případě již velice složité. Příkladem takovéto architektury je neuronová síť *ResNet* (He et al. 2016).

Autoři v tomto článku též zmiňují, že pro urychlení procesu vyhledání architektury sítě je vhodné stanovit základní blok vrstev, který bude v síti několikrát. Stanovení základního bloku významně redukuje množství kombinací, které by jinak musely být prohledány. Využití bloku jako základního stavebního prvku sítě též usnadňuje adaptaci dané architektury na jiná vstupní data (Zoph et al. 2018).

2.4.6 Konvoluční neuronová síť

Tento typ neuronových sítí je specifickým typem dopředných neuronových sítí, které obsahují alespoň jednu konvoluční vrstvu. První schéma konvoluční neuronové sítě (*convolution neural network – CNN*) navrhl Fukushima (1980). Tato konvoluční neuronová síť však ještě nevyužívala zpětné propagace. První konvoluční neuronovou síť, vycházející z fungování dopředných neuronových sítí včetně zpětné propagace byla navržena autory LeCun et al. (1999) až na přelomu tisíciletí. Konvoluční neuronové sítě (dále CNN) byly navrženy s cílem lépe rozeznávat různé vzory v obraze.



Obrázek 17: Architektura konvoluční neuronové sítě
***Zdroj**:* Deshpande (2016)

V CNN tvoří první vrstvu vždy konvoluční vrstva, obvykle následovaná sekvencí dalších konvolučních a *Max Pooling* vrstev. Každá CNN též obsahuje alespoň jednu *Fully-conected* vrstvu. Převod 2D výstupu z konvolučních a *Max Pooling* vrstev do 1D tak, aby mohl být zpracován pomocí *Fully-conected* vrstvy, obstarává nejčastěji již zmíněná *Flattening* vrstva nebo GAP vrstva. Typický příklad architektury CNN lze vidět na obrázku č. 17.

2.5 Metody pro hledání optimálních hyperparametrů

Cílem metody pro optimalizaci hyperparametrů, v oblasti strojového učení, je najít hyperparametry daného klasifikátoru, které poskytnou nejvyšší výkonost, např. klasifikační přesnost. Tento proces lze přeneseně označit jako hledání ideálního nastavení daného klasifikátoru.

1. **Manuální vyhledávání**

   Při tomto základním způsobu, jsou hodnoty hyperparametrů určeny ručně na základě expertních znalostí a metody pokus omyl. Tento postup lze aplikovat pouze příkladě jednodušších modelů s malým počtem hyperparametrů. V takovém to případě může být manuální vyhledávání relativně rychlým a účinným nástrojem. Manuální vyhledávání se však promítá i do pokročilejších metod. Pomocí expertních znalostí je možné značně snížit počet prvků v prohledávaném prostoru a tím výrazně urychlit celý proces (Bergstra, Bengio 2012).

2. **Grid search**

   Tento algoritmus je zástupcem takzvaných *exhaustive search* algoritmů neboli algoritmů, které prohledávají kompletně celý prohledávaný prostor. Pro aplikaci tohoto algoritmu je nutné hodnoty jednotlivých hyperparametrů diskretizovat. Výsledný prohledávaný prostor je poté definován jako kartézský součin tohoto konečného počtu hodnot náležícím jednotlivým hyperparametrům (Verwaeren, Van Der Weeën, De Baets 2015). Výhodou tohoto algoritmu je, že s určitostí dokáže najít nejlepší možné řešení. Nevýhodou ovšem je, že toto řešení pochází z konečného množství hodnot vzniklých diskretizací původně často spojitých hodnot hyperparametrů. Vzhledem k tomu, že je prohledáván celý konečný prohledávaný prostor, jedná se o velmi časově náročný algoritmus. Tento algoritmus lze však relativně snadno paralelizovat (Bergstra, Bengio 2012).

3. **Random search**

   Na rozdíl od algoritmu *Grid search*, neprohledává tento algoritmus všechna možná řešení. Konečný počet prohledávání je v tomto případě nahrazen náhodným výběrem kombinací hodnot vstupních hyperparametrů. Z tohoto důvodu nemá algoritmus principiálně konečný počet kroků řešení, a tudíž je nutné stanovit maximální počet prohledávání. Vzhledem k tomu, že se jedná o náhodný výběr, není nutné hodnoty vstupních parametrů diskretizovat. (Bergstra, Bengio 2012). I přes použití menšího počtu prohledávání, které by byly nutné pro prohledání kompletně celého

prohledávaného prostoru v případě algoritmu *Grid search*, může tento algoritmus najít srovnatelné a mnohdy i lepší řešení. Takového výsledku je dosaženo zejména v případě, že výkonost klasifikátoru je ovlivněna pouze malou částí všech hyperparametrů (Wang et al. 2016). Oproti předchozímu algoritmu však nalezení nejlepšího řešení není zaručeno. Stejně jako algoritmus *Grid search*, lze i tento algoritmus paralelizovat. Úspěšnost tohoto algoritmu může být též zvýšena pomocí specifikace rozložení, ze kterého jsou náhodné výběry prováděny.

4. **Bayesian optimization**

Tento algoritmus je jednou z globálních optimalizačních technik pro funkce tzv. **černých skříněk**. Neboli funkcí, u kterých známe vstupy a výstupy, avšak přesně neznáme, jak se daná funkce chová pro různé vstupy (Wu et al. 2019). Na rozdíl od dvou předchozích metod, využívá tento algoritmus záznamů o výkonnosti klasifikátoru při různých kombinacích hodnot vstupních hyperparametrů. Tyto hodnoty jsou použity pro vytvoření a průběžnou úpravu tzv. **náhradního modelu** (*surrogate model*). Cílem tohoto pravděpodobnostního modelu je napodobit chování modelu (funkce), jehož hyperparametry optimalizujeme. Na základě předchozích iterací, se tento náhradní model snaží předpovědět kombinaci hodnot vstupních hyperparametrů, která dosáhne nejvyšší výkonnosti (Hutter, Lücke, Schmidt-Thieme 2015). Několik výzkumů potvrdilo, že tento algoritmus dokáže nalézt lepší řešení při menším počtu vyzkoušených kombinací v porovnání s algoritmy *Grid search* a *Random Search*, zejména díky schopnosti učit se z již vyzkoušených kombinací (Hutter, Hoos, Leyton-Brown 2011),(Snoek, Larochelle, Adams 2012).

2.6 Metody pro výběr nejdůležitějších atributů

Cílem této kapitoly je poskytnout teoretický základ k metodám pro výběr nejdůležitějších atributů. Metody pro výběr atributů (*feature selection methods*) poskytují možnost, jak redukovat výpočetní čas, zvýšit přesnost klasifikace a též pomáhají lépe pochopit informace obsažené v datech (Chandrashekar, Sahin 2014).

Metody pro výběr atributů lze rozdělit do dvou základních skupin. **Obalové metody** (*Wrapper methods*) při výběru atributů vytváří velké množství modelů s různými výběry atributů. Konečný výběr atributů je poté založen na klasifikačním modelu s nejvyšším výkonem, např. nejvyšší klasifikační přesnost (Kuhn, Johnson 2013). Příkladem tohoto

typu je RFE (*Recursive feature elimination*)(Guyon et al. 2002). Výraznou nevýhodou tohoto typu metod je velká časová náročnost (Chandrashekar, Sahin 2014).

Druhou skupinou jsou tzv. **filtrační metody** (*Filter methods*). Tyto metody využívají statistických technik pro ohodnocení vztahu mezi jednotlivými atributy a predikovanými proměnnými. V závislosti na hodnotě určité statistické proměnné je poté daný atribut vybrán či nevybrán pro vstup do klasifikačního modelu (klasifikátoru). Dále jsou podrobněji popsány tři metody pro výběr atributů, které jsou využity dále v práci. Všechny dále zmíněné metody jsou představiteli filtračních metod.

1. **Výběr atributů na základě korelace**

   Tato metoda vychází z předpokladu, že atributy s vysokou korelací jsou více lineárně závislé, a tudíž mají téměř stejný efekt na výsledný klasifikovaný typ. Tudíž, pokud dva atributy spolu vysoce korelují, je možné využít pouze jeden z nich (Guyon, André 2003). První krokem pro výběr atributů je vytvoření matice korelačních koeficientů. Konkrétně je pro každou kombinaci vstupních atributů spočtena hodnota *Pearsonova korelačního koeficientu*. V dalším kroku jsou všechny atributy přidány do finálního seznamu atributů. Následně jsou postupně procházeny všechny hodnoty korelačního koeficientu nacházející se pod diagonálou. V případě, že absolutní hodnota korelačního koeficientu pro danou kombinaci atributů je vyšší než stanovená mez, je jeden z atributů odebrán z finálního seznamu atributů.

2. **Výběr atributů na základě ANOVA testu**

   Tato metoda používá pro posouzení vlivu jednotlivých atributů na hodnotu výsledné proměnné **F-hodnotu**. Tato hodnota udává poměr mezi vysvětleným a nevysvětleným rozptylem. Zjednodušeně řečeno, F-hodnota udává jaké množství informace nese každý z atributů (Elssied, Ibrahim, Osman 2014). Pro výpočet F-hodnot je využita analýza rozptylu (ANOVA). Do seznamu finálních atributů poté je přidán požadovaný počet atributů s nejvyšší F-hodnotou (Arowolo et al. 2016).

3. **Transformace vstupních atributů pomocí metody PCA**

   Tato metoda využívá pro snížení počtu dimenzí vstupních dat **metodu hlavních komponent** (PCA). PCA představuje lineární kombinaci všech vstupních atributů na nové, vzájemně nekorelované proměnné (hlavní komponenty). Jinak řečeno, na rozdíl od předchozích dvou metod, PCA nevybírá nejvhodnější atributy, ale transformuje informace v nich obsažených do nově vytvořených atributů (komponent) (Song, Guo, Mei 2010). Výsledné hlavní komponenty jsou seřazeny podle velikosti rozptylu, který

je každou z komponent vysvětlený. Redukce dimenzí pomocí této metody je poté založena na předpokladu, že pomocí několika prvních komponent je vysvětlena většina rozptylu obsaženého v původních datech, a tudíž těchto několik hlavních komponent nese srovnatelné množství informace jako původní data (Guo et al. 2002).

2.7 Metriky pro hodnocení klasifikace

Jednou z nejčastěji používaných evaluačních metod výkonosti klasifikátorů je **matice záměn** (*Confusion matrix*). Matice záměn neboli chybová matice je specifickým případem kontingenční matice, která se používá pro přehlednou vizualizaci vztahu dvou nebo více statistických znaků (Zvára, Anděl, Martinková 2013). Každý řádek v matici představuje počet skutečných instancí dané třídě, každý sloupec poté počet predikovaných instancí dané třídy. Správné predikce se poté nacházejí na diagonále dané matice. Ukázku matice záměn binárního klasifikátoru lze vidět na obrázku č. 18. V případě binárního klasifikátoru mohou nastat 4 možnosti. První možnost **TN** (*true negative*) představuje počet správně klasifikovaných negativních hodnot (hodnota klasifikátoru 0). Druhá možnost **TP** (*true positive*) představuje počet správně klasifikovaných pozitivních hodnot (hodnota klasifikátoru 1). **FN** (*false negative*) představuje počet chybně klasifikovaných pozitivních hodnot. Poslední možnost **FP** (*false positive*) představuje počet chybně klasifikovaných negativních hodnot.

		Predikované	
	N = 100	**0**	**1**
Skutečné	**0**	TN = 35	FP = 5
	1	FN = 15	TP = 45

Obrázek 18: Matice záměn
***Zdroj**: vlastní zpracování*

Matice záměn však není vhodná pouze k vizuálnímu posouzení výkonosti klasifikátoru. Z hodnot uvnitř chybové matice lze vypočíst několik kvantitativních charakteristik. První metrikou charakterizující celkovou výkonnost klasifikátoru je **celková přesnos**t (*accuracy)*. Celková (klasifikační) přesnost je definována jako poměr počtu správně predikovaných prvků ku celkovému počtu prvků. Pro binární klasifikátor lze výpočet celkové přesnosti zapsat vzorcem:

$$accuracy = \frac{TN + TP}{TN + FN + TP + FP}.$$

Obecně platí, že celková přesnost je rovna poměru počtu prvků na diagonále ku počtu všech prvků v chybové matici. Použití celkové přesnosti pro hodnocení klasifikace je vhodné v případě, že počty prvků v jednotlivých třídách jsou přibližně stejné. Pro určení výkonosti klasifikátoru z pohledu jednotlivých tříd se využívají tři základní charakteristiky. První metrikou je **zpracovatelská přesnost** (*precision*). Hodnota této metriky je závislá pouze na počtu prvků predikovaných jako pozitivní. Matematicky lze zpracovatelskou přesnost vyjádřit jako:

$$precision = \frac{TP}{TP + FP}.$$

V případě klasifikace více tříd je tato metrika rovna poměru počtu správně predikovaných prvků dané třídy ku všem prvkům predikovaných jako daná třída. Zpracovatelská přesnost je vhodným ukazatelem v případě, že cena za špatně klasifikovaný pozitivní prvek je vysoká. Metrikou založenou na počtu prvků predikovaných jako negativní je **uživatelská přesnost** (*recall*). Tuto metriku lze pomocí vzorce vyjádřit jako:

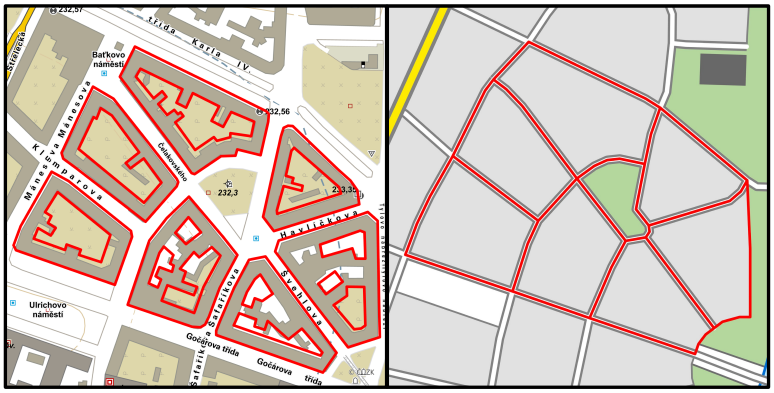
$$recall = \frac{TN}{TN + FN}.$$

V případě klasifikace více tříd se jedná o poměr správně predikovaných negativních prvků ku počtu všech prvků predikovaných jako negativní. Uživatelská přesnost je tedy vhodným ukazatelem v případě, že cena za chybně klasifikovaný negativní prvek je vysoká. Poslední metrika *F-score* kombinuje hodnotu dvou předchozích charakteristik do jediné. Jejím cílem je pomocí jediné hodnoty popsat výkonost klasifikátoru pro každou ze tříd. Její výpočet je shodný jak pro binární klasifikace, tak pro klasifikaci více tříd. Konkrétně je tato metrika definována jako:

$$F - score = 2 * \frac{precision * recall}{precision + recall}.$$

3 Typologie zástavby

Cílem této kapitoly práce je představit vlastní navrženou typologii zástavby. Inspirací pro vlastní navrženou typologii zástavby byly zejména typologie představené autory Du et al. (2016), Zhang et al. (2013), Hecht et al. (2013) a Vanderhaegen, Canters (2010). Typologie představené zmíněnými autory bylo však nutné upravit tak, aby navržená typologie více reflektovala zástavbu na území Česka. Zároveň byl při tvorbě typologie zástavby kladen důraz na propojení jednotlivých typů a metod pro generalizaci zástavby užívaných pro tvorbu topografické mapy v měřítku přibližně 1 : 50 000. Cílem propojení mezi navrženými typy zástavby a kartografickými reprezentacemi zástavby na ZM 50 je využití dat ze ZM 50 pro hodnocení přesnosti klasifikace na větším celku dat. Toto hodnocení je provedeno v kapitole č. 5.2. Celkem bylo pro účely této diplomové práce vymezeno šest typů zástavby. Charakteristika jednotlivých typů, včetně vazby na způsob znázornění daného typu zástavby na ZM 50, je popsána dále v této kapitole.



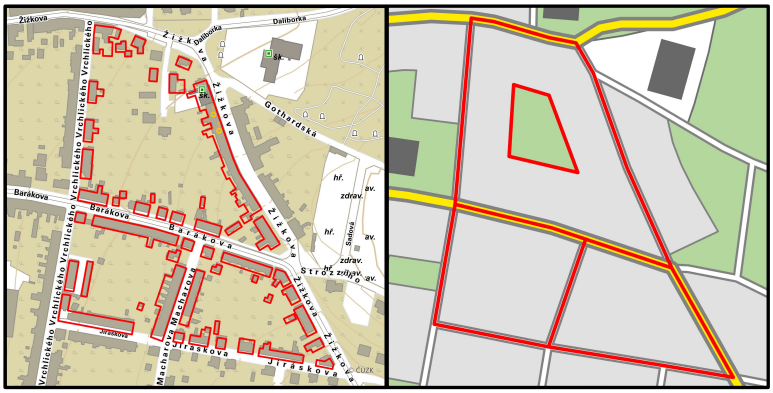
Obrázek 19: Typ zástavby Blok, Vlevo: ZABAGED, Vpravo: ZM 50
***Zdroj**: ČÚZK (2020a), ČÚZK (2020d), upraveno*

1. **Blok**

Prvním vymezeným typem zástavby je *Blok*. Tento typ zástavby se nejčastěji nachází v historických centrech měst. Shluky budov tohoto typu jsou velice často tvořeny pouze jednou budovou, ve smyslu jednoho polygonu objektu typu *Budova, blok budov* v ZABAGED (ČÚZK 2018) (obrázek 19, vlevo), popřípadě více budovami těsně navazujícími na sebe. Uvnitř shluku budov tohoto typu se může vyskytovat vnitroblok. Plocha vnitrobloku by však neměla přesáhnout plochu budov obklopující daný vnitroblok. Budovy tohoto typu zástavby jsou obvykle tvořeny čtyřmi a více patry (Hecht et al. 2013). Pro zobrazení tohoto typu zástavby na mapě v měřítku 1 : 50 000 je nejčastěji využívána generalizační metoda agregace. Na ZM 50 je tento typ zástavby znázorňován pomocí světle šedých polygonů (obrázek 19, vpravo).

2. **Blok s vnitroblokem**

Dalším vymezeným typem je *Blok s vnitroblokem*. Tento typ zástavby je často situován v centrech menších měst, popřípadě na okraji větších měst. Oproti předchozímu typu, je tento typ tvořen převážně řadovými domy, které již nedosahují takového počtu pater jako první typ zástavby. Řadové domy jsou nejčastěji situovány podél uliční sítě a dohromady vymezují prostor uzavřený mezi nimi (obrázek 20, vlevo). Plocha, kterou zaujímá tento uzavřený prostor (vnitroblok) je v tomto případě mnohem větší než plocha budov, které tvoří daný shluk budov (blok). Jak lze vidět na ukázce na obrázku č. 20, oproti předchozímu typu se mezi jednotlivými budovami mohou častěji vyskytovat mezery. Pro zobrazení tohoto typu zástavby na mapě v měřítku 1 : 50 000 je nejčastěji využívána generalizační metoda agregace. V závislosti na velikosti celého bloku je vnitroblok zachován či spojen společně s budovami. V případě, že budovy po obvodu tohoto bloku nenavazují těsně na sebe, lze využít též generalizační metodu typifikace. Na ZM 50 je tento typ zástavby nejčastěji znázorněn světle šedým polygonem bez zachování vnitrobloku (obrázek 20, vpravo).

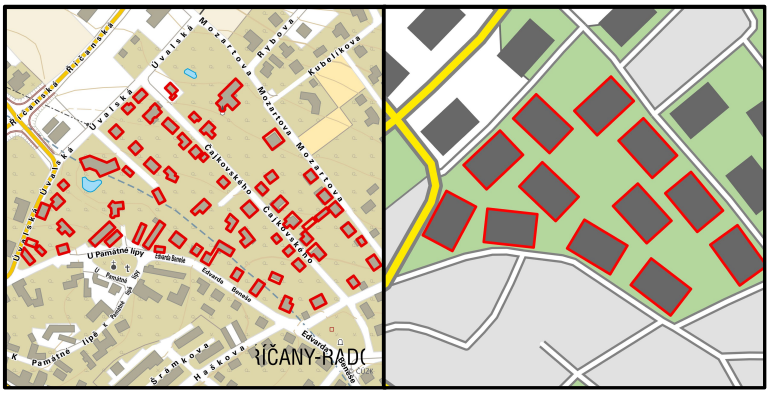


Obrázek 20: Typ zástavby Blok s vnitroblokem, Vlevo: ZABAGED, Vpravo: ZM 50
***Zdroj**: ČÚZK (2020a), ČÚZK (2020d), upraveno*

3. **Malé a střední domy**

Třetím vymezeným typem zástavby jsou *Malé a střední domy*. Tento typ zástavby se nejčastěji nachází na okraji měst. Budovy, které tvoří shluky budov tohoto typu jsou představovány zejména rodinnými domy (jedno i dvougenerační domy), které jsou obvykle tvořeny dvěma patry (Hecht et al. 2013). Jednotlivé budovy mohou být ve shluku uspořádány do pravidelného tvaru, například mřížky. Takto pravidelnou zástavbu lze nalézt zejména v suburbánní zástavbě. Budovy ve shluku mohou být též neuspořádané (obrázek 21, vlevo). Pro zobrazení tohoto typu zástavby na mapě v měřítku 1 : 50 000 je nejčastěji využívána generalizační metoda typifikace. V případě, že nejsou jednotlivé budovy příliš vzdálené je možné uplatnit též agregaci. Prakticky se

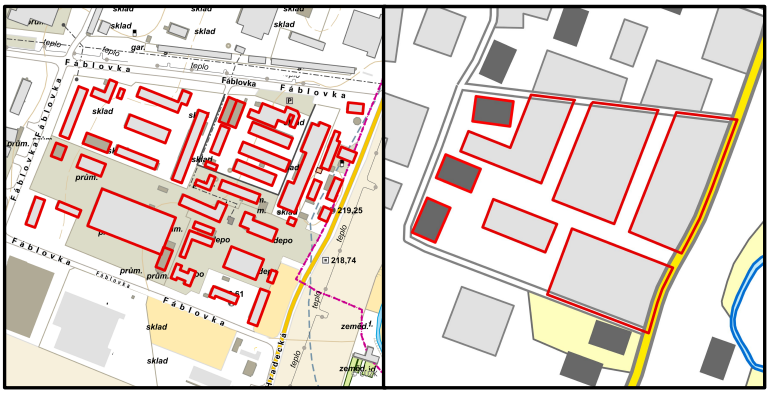
generalizace na ZM 50 provádí pomocí bodového znaku, reprezentovaného tmavě šedým obdélníkem s pevnou velikostí a proměnlivou orientací (obrázek 21, vpravo). V případě, že je použita metoda agregace, je daný typ znázorněn pomocí světle šedého polygonu, stejně jako předchozí dva typy zástavby.



Obrázek 21: Typ zástavby Malé a střední domy, Vlevo: ZABAGED, Vpravo: ZM 50
***Zdroj**: ČÚZK (2020a), ČÚZK (2020d), upraveno*

4. **Industriální zástavba**

Dalším vymezeným typem je *Industriální zástavba*. Tento typ zástavby je tvořen převážně plošně rozlehlými budovami, např. velkými továrními halami, často doplněnými o řadu menších budov (obrázek 22, vlevo). Do daného typu lze též zařadit např. budovy nemocnic či škol. Pro zobrazení tohoto typu zástavby na mapě v měřítku 1 : 50 000 je nejčastěji využívána generalizační metoda zjednodušení (pro velké objekty) a agregace (pro menší objekty). Velké budovy náležící do tohoto typu zástavby jsou na ZM 50 znázorněny pomocí světle šedých polygonů. Menší budovy jsou znázorněny pomocí bodového znaku představovaného tmavě šedým obdélníkem (obrázek 22, vpravo).

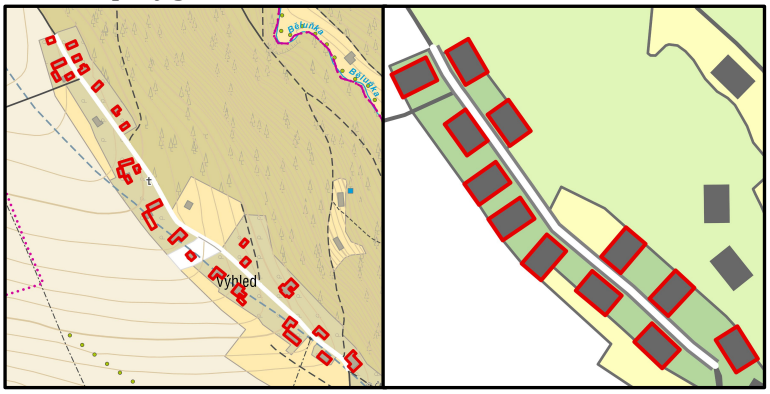


Obrázek 22: Typ Industriální zástavba, Vlevo: ZABAGED, Vpravo: ZM 50
***Zdroj**: ČÚZK (2020a), ČÚZK (2020d), upraveno*

5. **Liniová zástavba**

Předposledním vymezeným typem je *Liniová zástavba*. Tento typ zástavby lze nejčastěji nalézt v menších městech a vesnicích. Obdobně jako v případě typu *Malé*

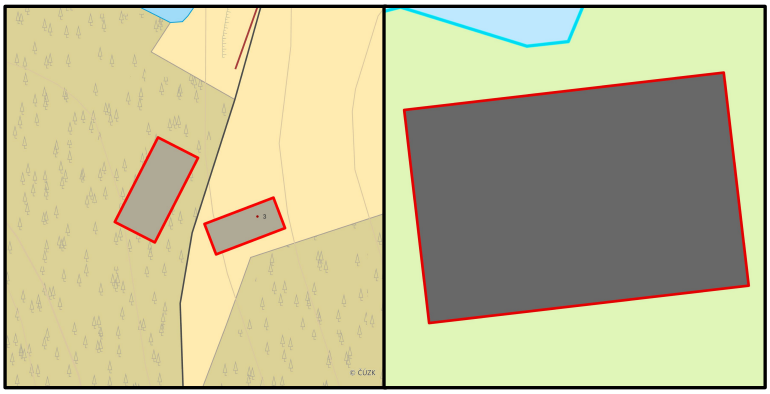
a střední domy tvoří tento typ převážně rodinné domy. V tomto případě jsou však budovy rozmístěny podél silniční sítě (obrázek 23, vlevo). Pro generalizaci tohoto shluku se též nejčastěji využívá typifikace. Při typifikace *Liniové zástavby* hraje velice důležitou roli silniční síť, neboť velice často typifikované budovy mají stejnou orientaci, odpovídající přiléhající silnici. Kartografickou reprezentací na ZM 50 je podobně jako u *Malých a střední domů* bodový znak, reprezentovaný tmavě šedým polygonem (obrázek 23, vpravo). V případě, že budovy v liniové zástavbě přiléhají těsně k sobě, mohou být též při generalizaci agregovány a na ZM 50 znázorněny pomocí světle šedého polygonu.



Obrázek 23: Typ Liniová zástavba, Vlevo: Zabaged, Vpravo: ZM 50
***Zdroj**: ČÚZK (2020a), ČÚZK (2020d), upraveno*

6. Samota

Posledním vymezeným typem zástavby je *Samota*. Tento typ zástavby je nejčastěji tvořen jednou, či několika málo budovami (obrázek 24, vlevo). Tyto budovy jsou nejčastěji tvořeny rodinným domem. Při generalizaci tohoto typu shluku budov do měřítka 1 : 50 000 dochází ke vzniku jediné typifikované budovy. Na ZM 50 je tento typ zástavby znázorněn pomocí jediného bodového znaku, reprezentovaného tmavě šedým obdélníkem (obrázek 24, vpravo)



Obrázek 24: Typ zástavby Samota, Vlevo: ZABAGED, Vpravo: ZM 50
***Zdroj**: ČÚZK (2020a), ČÚZK (2020d), upraveno*

Z definic jednotlivých typů vyplývá, že představené typy nemají přesně definované hranice, a tudíž může docházet k jejich překryvu. Zejména rozdíl mezi typy 2, 3 a 5 je relativně malý. Pro generalizaci všech těchto tří typů je teoreticky možné použít typifikaci. Cílem podrobnějšího dělení bylo však zdůraznit jevy, které budou silně ovlivňovat generalizaci daného typu. V případě typu *Blok s vnitroblokem* jsou těmito jevy uliční síť, podél níž jsou budovy tohoto typu orientované, a prostor uzavřený shlukem budov tohoto typu. Pro typ *Malé a střední domy* je nejdůležitějším jevem vnitřní struktura shluku budov, která by při typifikaci měla být zachována, zejména v případě pravidelného vzoru, jako je mřížka. Uliční síť v případě typu *Malé a střední domy* již nehraje významnou roli. Typ *Liniová zástavba* lze považovat za speciální případ typu *Blok s vnitroblokem*. V případě *Liniové zástavby* je však hlavním prvkem ovlivňující proces generalizace pouze silniční síť.

4 Metodika

Cílem této kapitoly práce je představit algoritmus pro klasifikaci zástavby pro účely kartografické generalizace pomocí tří různých klasifikátorů. Celý algoritmus lze připodobnit k objektově orientované klasifikaci obrazu využívané v dálkovém průzkumu země (DPZ) pro klasifikaci satelitních snímků. První fáze algoritmu je představována segmentací, při níž jsou hledány shluky budov. Vlastní klasifikace je poté provedena pomocí příznaků (vlastností) jednotlivých shluků.

V podkapitole 4.1 jsou představeny data vstupující do algoritmu a též skutečné výstupy. Následující podkapitola 4.2 je věnována softwaru, který byl použit ke zpracování dat, vlastním výpočtům a hodnocení výsledků. V podkapitole 4.3 je následně podrobněji popsán navržený proces segmentace. Podkapitola 4.4 a 4.5 je věnována přípravě dat vstupujících do jednotlivých klasifikátorů. V následující podkapitole 4.6 je popsána implementace metod pro segmentace a přípravy dat včetně diskuse konkrétních hodnot parametrů zmíněných v kapitolách 4.3 až 4.5. Na závěr je popsán postup klasifikace zástavby pomocí tří různých klasifikátorů, včetně dosažených výsledků.

4.1 Data

Nejdůležitějšími vstupními daty jsou polygony budov ze *ZABAGED* (Základní báze geografických dat)(ČÚZK 2020d), které mi byly poskytnuty pro účely této diplomové práce *Českým úřadem zeměměřickým a katastrálním* (ČÚZK). Další vstupní data pocházejí z volně dostupného zdroje *Data50* (ČÚZK 2020a). Tato datová sada představuje vektorovou podobu *ZM 50* (Základní mapa 1 : 50 000)(ČÚZK 2020e). Z teoretické části práce vyplývá, že generalizace zástavby je silně ovlivněna liniovými prvky, jejichž generalizace předchází generalizaci zástavby (Kilpelainen 1995)(Gottstein 2019). Z tohoto důvodu, byla pro segmentaci budov do shluků použita právě tato datová sada obsahující již generalizované liniové objekty. Konkrétně byly využity:

- cesta,
- silnice a dálnice,
- ulice,
- železniční trať,
- vodní tok.

Z téže datové sady-*Data50* byly též využity dvě vrstvy reprezentující zástavbu. První vrstva je tvořena polygony, které ve značkovém klíči ZM 50 představují objekt *Blok*

budov. Druhá vrstva je bodová. Tato bodová vrstva ve značkovém klíči ZM 50 představuje objekt *Budova*. Tyto dvě vrstvy jsou využity pro ověření přesnosti klasifikace zástavby v kapitole 5.2. Poslední datové zdroje využité v této práci jsou představovány digitálním modelem reliéfu (DMR) a digitálním modelem povrchu (DMP). Rozdíl těchto modelů lze využít pro přibližné určení výšky budov. Konkrétně se jedná o *Digitální model reliéfu České republiky 5. generace* (DMR 5G) a *Digitální model povrchu České republiky 1. generace* (DMP 1G). K oběma modelům lze volně přistupovat pomocí *IMAGE* služby *Esri ArcGIS Server*. Jedná se o online službu poskytovanou ČÚZK. V této podobě jsou oba modely reprezentovány jako rastr s prostorovým rozlišením 2 m (ČÚZK 2020c; 2020b).

Výstupní data jsou tvořena polygonovou vrstvou budov včetně několika atributů (příloha 2). První z atributů *Cluster_ID* obsahuje hodnotu jedinečného identifikátoru shluku budov. Tímto je každá budova jednoznačně přiřazena ke konkrétnímu shluku. Následující atributy udávají nejpravděpodobnější typ zástavby určený konkrétním klasifikátorem, včetně určení pravděpodobnosti zařazení daného shluku k jednotlivým typům zástavby.

4.2 Software

Aby bylo možné veškeré výpočty provádět opakovaně a nad rozsáhlejšími daty, byla celá metodická část práce napsána jako soubor několika skriptů v jazyce *Python* (příloha 1). Pro výpočty a geometrické operace týkající se segmentace a výpočtu charakteristik shluků byla využita zejména knihovna *ArcPy*, která zpřístupňuje funkcionalitu softwaru *ArcGIS Pro* pro skriptování (ESRI 2020). S využitím knihovny *Psycopg* (Di Gregorio 2020) mohly být některé charakteristiky shluků budov počítány pomocí databázové nadstavby *PostGIS*. Díky tomu byl výrazně zkrácen čas pro výpočet daných charakteristik.

Pro navazující výpočty klasifikace zástavby byla využita služba *Google Colab*. Jedná se online službu, která umožňuje v rámci vývojového prostředí *Jupyter Notebooks* spouštět skripty v jazyce *Python* na výpočetních jednotkách společnosti *Google*. Největší výhoda této služby spočívá ve volnému přístupu k *GPU* (graphics processing unit)(Google 2020). GPU jsou výhodná zejména při práci s neuronovými sítěmi, neboť výrazně urychlují proces učení (Microsoft 2020). Vlastní klasifikace pomocí neuronových sítí byla naprogramována s využitím knihovny *Tensorflow*. Jedná se o open-source knihovnu od společnosti Google, která umožňuje, díky velké míře abstrakce, relativně snadnou

implementaci neuronových sítí (Abadi et al. 2016). K nalezení ideálních hyperparametrů neuronových sítí byla využita knihovna *Optuna*, která je založena na bayesovské optimalizaci (Akiba et al. 2019). Pro klasifikaci zástavby pomocí logistické regrese byla využita knihovna pro strojové učení *Scikit-learn* (Pedregosa et al. 2011). Tato knihovna obsahuje velké množství klasifikačních, regresních a shlukovacích algoritmů.

4.3 Segmentace

Na základě rešerše metod pro segmentaci budov do shluků v kapitole 2.2, byl vytvořen algoritmus kombinující segmentaci pomocí liniových prvků a metody DBSCAN (Ester et al. 1996)(Basaraner, Selcuk 2008), které byly blíže popsány v kapitole 2.2. Generalizovaná cestní síť a další liniové prvky dokážou velmi dobře segmentovat budovy do shluků v místech s hustou zástavbou a s hustou cestní sítí. Na ostatních místech dochází ke vzniku příliš velkých shluků budov. Proto byla metoda segmentace pomocí cestní sítě doplněna o metodu DBSCAN, která pracuje s vzdáleností jednotlivých budov. Výsledkem procesu segmentace jsou polygony s doplněným atributem, který určuje příslušnost budovy ke konkrétnímu shluku. Vlastní proces segmentace lze tedy rozdělit do tří základních kroků:

1. segmentace pomocí liniových bariér,
2. aplikace metody DBSCAN na již vytvořené shluky,
3. kontrola protnutí s cestní sítí.

4.3.1 Segmentace pomocí liniových bariér

Cílem tohoto kroku je rozdělit celé zájmové území pomocí liniových bariér na množství menších polygonů a následně určit příslušnost každé budovy k jednotlivým vzniklým polygonům. Níže jsou popsány jednotlivé kroky, které jsou nutné k dosažení tohoto cíle, a to včetně názorných obrázků.

1. **Vytvoření konvexní obálky**

   Během tohoto kroku je vytvořen polygon ohraničující celé zájmové území v podobě konvexní obálky okolo všech budov. Tento polygon je během následujících kroků rozdělen pomocí liniových bariér (obrázek 25, modře).

2. **Spojení vstupních linií do jediné vrstvy**

   V tomto kroku je nejdříve převedena vytvořená konvexní obálka na linii a následně je spojena se všemi liniovými prvky (bariérami) popsanými v kapitole 4.1 do jediné vrstvy liniových bariér. Vrstvu linií lze vidět na obrázku č. 25.

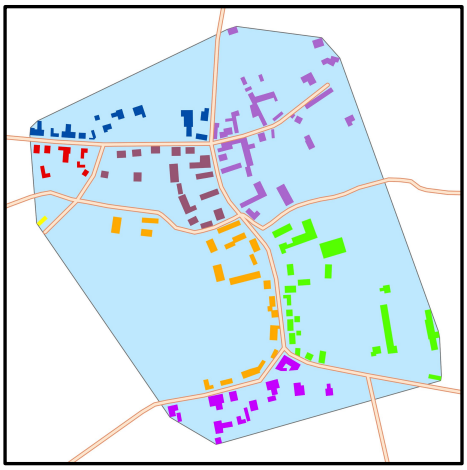
3. **Rozdělení polygonu pomocí linií**

Zde dochází k převedení areálů kompletně ohraničených liniemi z předchozího kroku zpět na polygon. Vzhledem k tomu, že jedna z linií tvoří hranici konvexní obálky okolo všech budov, je zaručeno, že každá vstupní budova je překryta alespoň jedním ze vzniklých polygonů. Nově vzniklým polygonům jsou přiřazeny unikátní identifikátory.

Obrázek 25: Dělící linie pro segmentaci včetně konvexní obálky (modře)
***Zdroj**: ČÚZK (2020a), ČÚZK (2020d), vlastní zpracování*

4. **Přiřazení budov ke vzniklým polygonům**

Jednotlivým budovám je poté přiřazen zmíněný unikátní identifikátor polygonu, v závislosti na tom, ve kterém polygonu leží centroid dané budovy. Shluky budov (budovy se stejným identifikátorem) barevně odlišené podle unikátních identifikátorů lze vidět na obrázku č. 26.



Obrázek 26: Shluky budov po segmentaci pomocí dělících linií
***Zdroj**: ČÚZK (2020a), ČÚZK (2020d), vlastní zpracování*

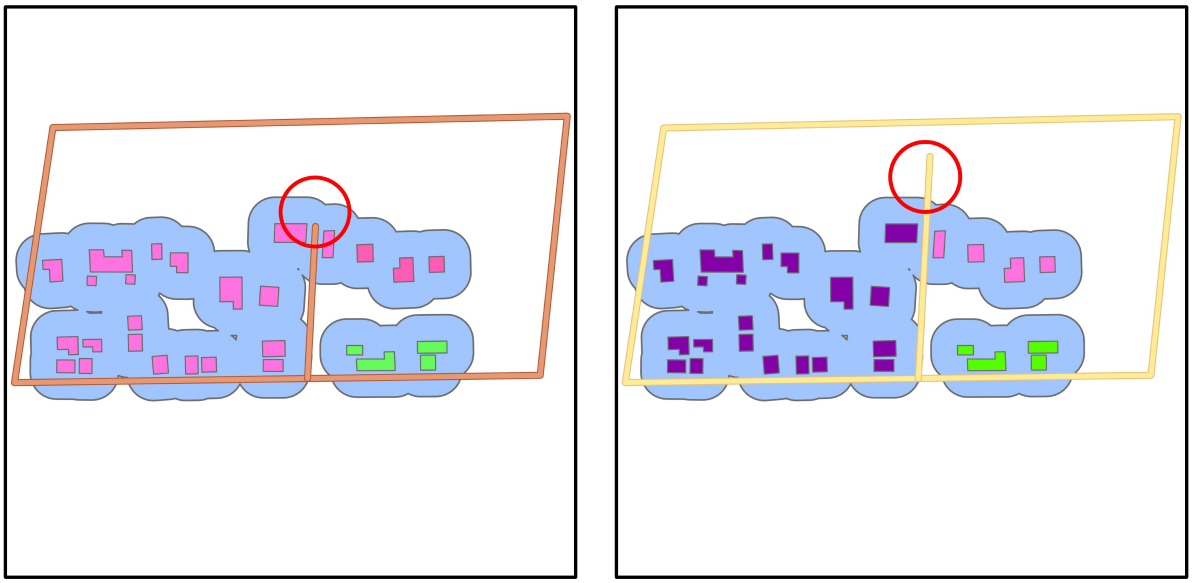
4.3.2 Aplikace metody DBSCAN na již vytvořené shluky

V předchozím kroku mohou v oblastech s řídkým výskytem liniových bariér vzniknout značně prostorově rozsáhlé a nehomogenní shluky budov, které by bylo

obtížné klasifikovat. Při podmínce, že pro zařazení budovy do shluku stačí, aby jediný soused byl blíže než zvolená vzdálenost (ε=1), lze metodu *DBSCAN* snadno implementovat pomocí bufferu okolo jednotlivých budov a následného hledání, zda se tyto buffery protínají.

V této části segmentace jsou postupně zpracovávány veškeré shluky budov vzniklé v předchozím kroku segmentace. S každým shlukem budov je v tomto kroku pracováno jako se samostatným objektem. Na každý shluk budov je aplikována metoda DBSCAN. Tato metoda může, ale nemusí daný shluk budov rozdělit.

Jako první je okolo každé budovy vytvořen buffer ve vzdálenosti *d*. Tyto buffery jsou následně spojeny, tak jak to lze vidět na obrázku č. 27, vlevo. Každému vzniklému bufferu je opět přiřazen unikátní identifikátor. Následně je budovám, náležícím do těchto spojených bufferů, přiřazen onen unikátní identifikátor. Na obrázku č. 27, vlevo, je původně jeden shluk budov, který vznikl po segmentaci pomocí liniových prvků. Po vytvoření bufferů je patrné, že vznikly dva nové shluky budov.



Obrázek 27: Segmentace metodou DBSCAN
Vlevo: shluky po aplikaci DBSCAN, Vpravo: Finální shluky
***Zdroj**: ČÚZK (2020a), ČÚZK (2020d), vlastní zpracování*

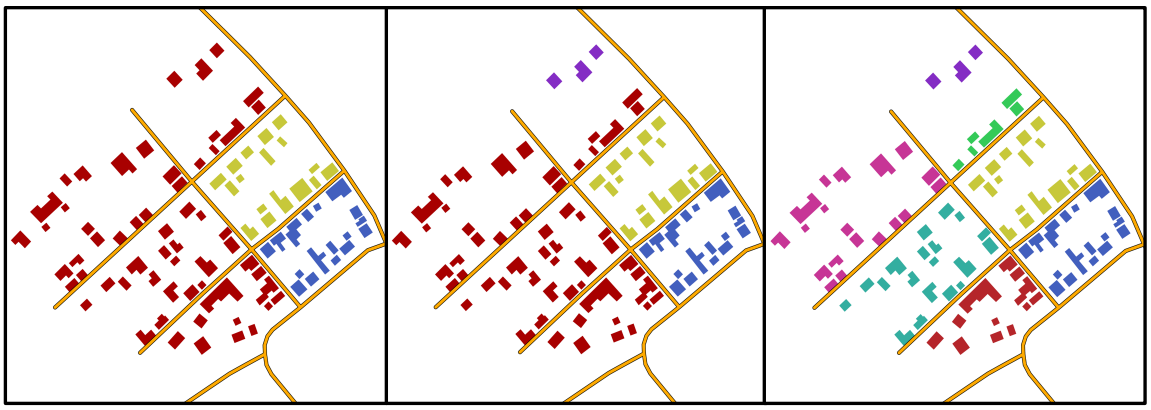
4.3.3 Kontrola protnutí s cestní sítí

V tomto kroku je pro každý shluk, vzniklý pomocí metody DBSCAN, hledán průnik bufferu z metody DBSCAN obklopující daný shluk budov s liniovými bariérami. Tento případ může nastat v místech, kde liniovou bariéru tvoří slepá ulice viz obrázek č. 27 (červený kruh). V tomto případě je nutné shluk budov dále rozdělit. Tento krok je prováděn i v případě, že metoda DBSCAN shluk budov vzniklý v prvním kroku

segmentace nijak nerozdělila. Pokud není žádný průnik nalezen, je daný shluk budov označen jako konečný.

V případě, že je průnik bufferu s liniovými bariérami potvrzen, jsou nejprve vybrány ty liniové bariéry, které daný průnik způsobují. Pokud se mezi vybranými liniemi nachází linie, která není na obou koncích napojena, je v místě volného konce prodloužena ve směru posledního úseku o vzdálenost *2d*. Toto prodloužení má zajistit, aby liniové bariéry nekončily uvnitř bufferu, ale protnuly ho (obrázek 27, vpravo). Na budovy uvnitř shluku budov, který je protnut liniovými bariérami je poté opět aplikována lehce upravená metoda představená v kapitole 4.3.1.

V tomto případě je tvorba konvexní obálky nahrazena již vytvořeným spojeným bufferem okolo budov. Obdobně jako ve zmíněné kapitole, je obvodová linie bufferu převedena na linii a spojena s vybranými liniovými bariérami. Stejně jako ve třetím kroku dané metody je spojená liniová vrstva převedena zpět na polygony. Tento krok zajistí rozdělení spojeného bufferu pomocí vybraných linií. Daným polygonům je opět přiřazen unikátní identifikátor. Posledním krokem druhé části segmentace je přiřazení onoho unikátního identifikátoru polygonu budovám. Tímto krokem je proces segmentace budov do shluků ukončen. Na obrázku č. 28 lze pozorovat výsledky jednotlivých kroků segmentace pomocí barevně odlišených shluků na stejných vstupních datech.



Obrázek 28: Porovnání jednotlivých kroků segmentace
Vlevo: po segmentaci pomocí cestní sítě, Uprostřed: po aplikaci metody DBSCAN, Vpravo: konečný stav
***Zdroj**:* ČÚZK (2020a), ČÚZK (2020d), *vlastní zpracování*

4.4 Kritéria výběru

V této kapitole práce budou popsány jednotlivé charakteristiky shluků budov, které jsou následně využity pro klasifikaci zástavby pomocí logistické regrese a neuronové sítě. Snahou bylo spočítat větší množství charakteristik pro jednotlivé shluky budov i přes

jejich možnou vysokou korelaci a následně využít metod pro výběr těch nejdůležitějších charakteristik, které budou vstupovat do procesu klasifikace.

Metody pro výběr nejvhodnějších atributů pro klasifikaci shluků byly představeny v rešeršní části práce v kapitole 2.6. Všechny charakteristiky uvedené v této kapitole jsou počítány vždy nad jedním konkrétním shlukem, bez ohledu na interakci jednotlivých shluků mezi sebou. Několik charakteristik shluků bylo převzato z publikací zmíněných v rešeršní části práce. Některé charakteristiky byly vytvořeny specificky za konkrétním účelem odlišení vybraných typů zástavby. U jiných byl však pouze předpoklad, že nesou informaci, která by mohla pomoci jednotlivé typy rozlišit.

Níže jsou charakteristiky rozděleny do tří skupin podle způsobu jejich výpočtu. V první skupině se nachází charakteristiky typické tím, že jsou nejprve počítány pro každou budovu uvnitř shluku zvlášť a následně agregovány pro celý shluk. Ve druhé skupině jsou charakteristiky, které vznikají nad shlukem jako celkem. Poslední skupinu tvoří vzdálenostní charakteristiky. Přehled všech charakteristik je pak uveden v tabulce č. 2.

4.4.1 Charakteristiky nad jednotlivými budovami

Příznaky z této kategorie jsou nejdříve počítány pro každou budovu zvlášť a následně je využito agregační funkce pro přenesení vlastností jednotlivých budov na celý shluk. Jako agregační funkce byly využity: součet, aritmetický průměr a směrodatná odchylka. Vzorce pro výpočet jednotlivých agregačních funkcí jsou:

$$\text{součet: } \sum_{i=1}^{n} x_i = x_1 + x_2 + x_3 + \cdots + x_n,$$

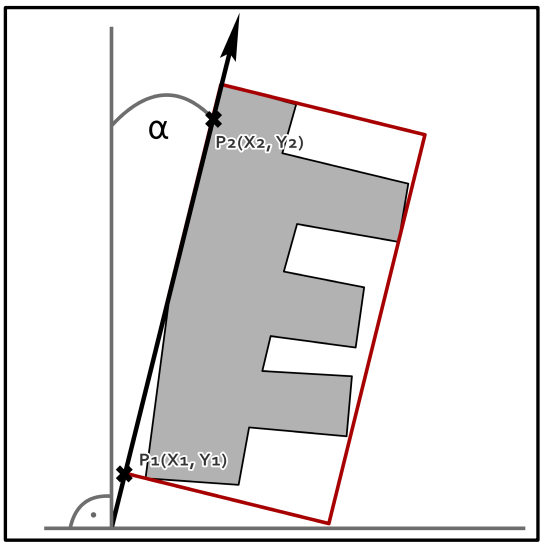
$$\text{aritmetický průměr: } \bar{x} = \frac{1}{n}(x_1 + x_2 + x_3 + \cdots + x_n) = \frac{1}{n}\sum_{i=1}^{n} x_i,$$

$$\text{směrodatná odchylka: } s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2},$$

kde x_i je hodnota příznaku na jedné budově a n je počet prvků shluku.

Základní charakteristikou shluku budov je **počet budov**. Nízký počet budov lze očekávat u typu zástavby *Blok* a zejména u typu *Samota*. Naopak vyšší počet budov lze očekávat u typu *Malé a střední domy.* Počet budov je též důležitý při výpočtu průměrů a směrodatných odchylek zbývajících charakteristik.

Následující tři charakteristiky byly odvozeny na základě **plochy jednotlivých budov**. Součet plochy budov lze považovat za jeden z ukazatelů velikosti shluku. Nevyšší sumu plochy budov lze očekávat u kategorie *Industriální zástavby*. **Průměr plochy jednotlivých budov** nám přináší informaci o typické velikosti budovy ve shluku. Vyšší průměrné hodnoty lze očekávat u kategorie *Blok*, nižší u typu *Malé a střední domy*. U kategorie *Industriální zástavba* není průměr díky zastoupení malých i velkých budov příliš vypovídající. Z tohoto důvodu byla spočtena i **směrodatná odchylka plochy budov**. Hodnota směrodatné odchylky vypovídá o homogenitě, respektive heterogenitě pozorované proměnné. Shluky budov s velice rozdílnou velikostí ploch jednotlivých budov budou vykazovat vysokou směrodatnou odchylku. Naopak ve shluku, kde budou všechny budovy velice podobně velké, bude hodnota směrodatné odchylky nízká. V našem případě se dá vysoká směrodatná odchylka očekávat u typu *Industriální zástavby*. Naopak kategorie *Malé a střední domy* bude charakteristická nižší směrodatnou odchylkou.

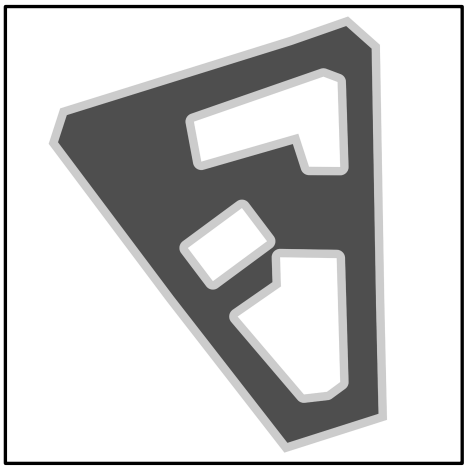


Obrázek 29: Výpočet azimutu budovy
***Zdroj**: ČÚZK (2020d), vlastní zpracování*

Pro jednotlivé budovy byla též spočtena **délka obvodové linie**. Na základě hodnot pro jednotlivé budovy byl vypočten **součet** a **průměr** pro daný blok. U těchto dvou charakteristik lze předpokládat vysokou korelaci s charakteristikami spočtenými na základě plochy budov. Z poměru plochy k obvodu budovy lze usuzovat tvar budovy. Čím více se tvar budovy blíží tvaru kruhu, tím bude tento poměr menší. Naopak u protáhlých budov bude tento poměr větší.

Cílem další charakteristiky je popsat rozdílné **natočení jednotlivých budov** uvnitř shluku. Zejména pro *Liniovou zástavbu* se očekává podobné natočení jednotlivých budov.

Pro popsání rozdílnosti natočení směru byla vybrána směrodatná odchylka. Pro určení směru natočení budovy byla zvolena metoda pomocí aproximace budovy pomocí opsaného obdélníku s minimální šířkou. Po vytvoření obdélníku je možné natočení budovy určit pomocí azimutu delší strany daného obdélníku (obrázek 29). Azimut přímky α lze poté určit při znalosti souřadnic dvou bodů $P_1(X_1, Y_1)$, $P_2(X_2, Y_2)$ (rohy obdélníku) určit ze vzorce: $\sin(\alpha) = \frac{|X_2 - X_1|}{|Y_2 - Y_1|}$.



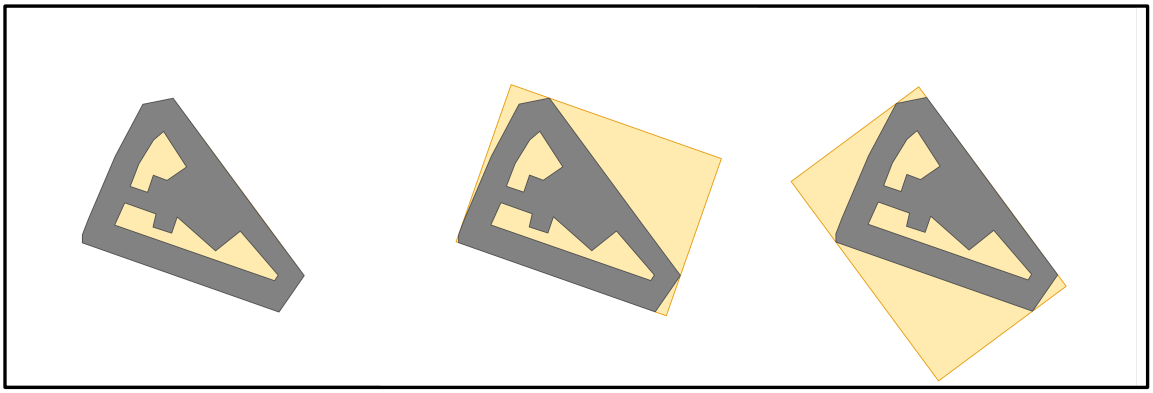
Obrázek 30: Polygon původní a zmenšené budovy pro výpočet průměrné výšky
***Zdroj**: ČÚZK (2020d), vlastní zpracování*

Poslední charakteristikou z popisované kategorie je **průměrná výška budov** ve shluku. Pro tuto charakteristiku se předpokládají nejvyšší hodnoty pro shluky typu *Blok*. Naopak kategorie *Malé a střední domy* tvořená převážně rodinnými domy, bude nejspíše dosahovat v průměru nižších výšek. Pro výpočet výšky budovy byly využity DMR a DMP. Z definic DMR a DMP plyne, že pokud tyto dva rastry od sebe odečteme, dostaneme nDMP, tedy rastr výšek všech objektů nad terénem. Před výpočtem průměrné výšky pro každou budovu byl nejprve zmenšen polygon každé budovy pomocí záporné hodnoty bufferu (obrázek 30). Cílem zmenšení plochy budov bylo opravit možné chyby plynoucí z nižšího rozlišení obou rastrů. Konkrétně byla snaha do výpočtu nezahrnout buňky rastru, které se nachází na okrajích budov, neboť hodnota těchto buněk nemusí odpovídat výšce budovy. Velikost zmenšení lze odvodit z faktu, že daná buňka rastru je do výpočtu výšky budov použita v případě, že je její střed obsažen v polygonu budovy. Z tohoto důvodu je nutné polygon zmenšit alespoň o velikost odpovídající ½ délky úhlopříčky dané buňky neboli přibližně o 75 % velikosti prostorového rozlišení. Pro výpočet výšky budovy byla následně využita zonální statistika. Jedná se o metodu, které pro každou zónu (polygon budovy) vrátí v našem případě průměrnou hodnotu všech

buněk rastru, jejichž střed se nachází v dané zóně (zmenšený polygon budovy). Pro výslednou hodnotu charakteristiky shluku byly výšky budov ve shluku zprůměrovány.

4.4.2 Charakteristiky nad shlukem jako celkem

Charakteristiky shluků pro jejich následnou klasifikaci jsou v této skupině založeny na tvorbě tří různých obalových geometrií, obklopujících shluk jako celek. Jako charakteristiky shluku jsou poté využity zejména vlastnosti daných geometrií jako je **plocha a obvod** nebo **šířka a výška** pro obdélníkovou geometrii. Zmíněné geometrie jsou představovány konvexní obálkou, opsaným obdélníkem s minimální šířkou a opsaným obdélníkem s minimální plochou. Jak již plyne z názvu kapitoly, tyto geometrie jsou tvořeny okolo celého shluku, nikoli okolo každé z budov ve shluku. Příklad tvaru těchto tří zmíněných geometrií na konkrétním shluku lze vidět na obrázku č. 31.



Obrázek 31: Porovnání obalových geometrií.
Vlevo: Konvexní obálka, Uprostřed: Opsaný obdélník s min. plochou, Vpravo: opsaný obdélník s min. šířkou
***Zdroj**: ČÚZK (2020d), vlastní zpracování*

Cílem charakteristik založených na těchto geometriích je popsat zejména **velikost a tvar** jednotlivých shluků. Jednotlivé charakteristiky z této kategorie nebyly vytvářeny s cílem odlišení dvou konkrétních informací, ale spíše jako zdroj dalších informací o shluku, který může klasifikátoru pomoct jednotlivé typy odlišit. Přehled všech charakteristik přímo plynoucích z vlastností těchto obalových geometrií, které byly využity pro tuto práci, lze nalézt v tabulce č. 2.

S využitím vlastností obalových geometrií byly spočteny dvě další charakteristiky. Účelem první charakteristiky je popsat **zastavěnost** daného shluku budov. Zastavěnost je představována poměrem rozlohy budov ku ploše, na které se rozprostírá daný shluk budov. Vyšší zastavěnost je očekávána u kategorie zástavby *Blok*, naopak u typu *Blok s vnitroblokem* je předpokládána nižší zastavěnost. Pro určení rozlohy celého shluku byla vybrána konvexní obálka, která ze zmíněních tří obalových geometrií nejlépe vystihuje velikost a tvar shluku.

Smyslem další charakteristiky je popsat **tvar shluku**. Konkrétně je cílem určit, zda se tvar bloku blíží více kruhu, čtverci či zda se jedná o protáhlejší tvar. Pro tuto charakteristiku bylo využito poměru šířky a délky shluku opsaného obdélníku s minimální šířkou. Tento příznak je důležitý pro odlišení *Liniového* typu zástavby.

Kategorie charakteristik	Charakteristika
Charakteristiky nad jednotlivými budovami	Počet budov
	Součet plochy budov
	Průměrná plocha budovy
	Směrodatná odchylka plochy budov
	Součet obvodů budov
	Průměrný obvod budovy
	Směrodatná odchylka orientace budovy
	Průměrná výška budovy
Charakteristiky nad shlukem jako celkem	Plocha konvexní obálky
	Obvod konvexní obálky
	Šířka opsaného obdélníku s minimální plochou
	Délka opsaného obdélníku s minimální plochou
	Plocha opsaného obdélníku s minimální plochou
	Šířka opsaného obdélníku s minimální šířkou
	Délka opsaného obdélníku s minimální šířkou
	Plocha opsaného obdélníku s minimální šířkou
	Poměr plochy budov ku ploše konvexní obálky
	Poměr šířky a délky opsaného obdélníku s minimální šířkou
	Poměr plochy budov k ploše polygonu z první části segmentace
Vzdálenostní charakteristiky	Průměrná vzdálenost k silnici
	Průměrná vzdálenost k nejbližší budově
	Průměrná vzdálenost ke třem nejbližším budovám

Tabulka 2: Kritéria výběru
***Zdroj**: Vlastní zpracování*

Poslední charakteristika v této kategorii již nevyužívá obalových geometrií. Jejím cílem je popsat **vliv cestní sítě** na tvar, respektive segmentaci shluku. Konkrétně byl využit poměr mezi plochou budov ve shluku a rozlohou polygonu, vzniklého během první části segmentace, příslušící danému shluku budov. Pokud je tato hodnota vysoká, lze předpokládat vysoký vliv cestní sítě. Tento jev lze očekávat zejména u zástavby typu *Blok*. Shluky budov tohoto typu vznikají nejčastěji již během prvního kroku segmentace a ve druhém kroku již nejsou dále děleny. Naopak nízké hodnoty uvedeného poměru lze očekávat u typu zástavby *Samota* či *Liniová zástavba*, protože konečná podoba shluků

těchto typů vzniká zejména až ve druhé části segmentace, neboť cestní síť nedokáže tyto typy dostatečně oddělit.

4.4.3 Vzdálenostní charakteristiky

V této kapitole budou představeny poslední tři charakteristiky vstupující do procesu klasifikace, které jsou založeny na výpočtu vzdáleností mezi každou budovou ve shluku a dalšími objekty.

První dvě charakteristiky pracují jen s budovami uvnitř daného shluku. Cílem obou těchto charakteristik je popsat rozložení budov uvnitř shluku. První charakteristika je založena na výpočtu **průměrné vzdálenosti k nejbližší budově v daném shluku**. Vzdálenost dvou polygonů je definována jako minimum vzdáleností mezi dvěma body takovými, že jeden leží na hranici jednoho polygonu a druhý na hranici druhého polygonu. Z definice segmentace plyne, že budovy nemohou být od sebe vzdáleny více než dvojnásobek šířky bufferu tvořeného okolo budov. Druhá charakteristika nepočítá **vzdálenost** pouze k nejbližší budově, ale **ke třem nejbližším budovám**. Přesněji jde o průměr z průměrů vzdáleností ke třem nejbližším budovám, tj. pro každou budovu shluku jsou nalezeny tři nejbližší sousedi. Vzdálenosti k těmto třem sousedům jsou následně zprůměrovány. Pro získání jediné hodnoty charakterizující cely shluk, jsou průměrné hodnoty vzdálenosti ke třem nejbližším sousedům pro každou budovu opět zprůměrovány. Rozdíl těchto dvou vzdáleností může pomoct pro odlišení *Liniového typu* zástavby a *Malých a středních domů*. U první zmíněné kategorie zástavby lze očekávat větší rozdíl mezi vzdáleností k nejbližší a ke třem nejbližším budovám. Oproti tomu lze u druhého zmíněného typu zástavby očekávat menší rozdíl mezi těmito dvěma hodnotami (obrázek 21 a 23).

Poslední charakteristika si opět klade za cíl odlišit zejména *Liniovou zástavbu* a *Malé a střední domy*. Pro jejich odlišení byla zvolena **průměrná vzdálenost budovy k cestní síti**. Vzhledem k tomu, že liniová zástavba se nejčastěji vyskytuje podél silnic, bude vzdálenost pro všechny budovy stejná a většinou i velmi malá. Naopak budovy v kategorii *Malé a střední domy* nebývají všechny situované přímo u cestní sítě. Proto lze očekávat vyšší hodnoty průměrné vzdálenosti k silnici (obrázek 21 a 23).

4.5 Tvorba rastrových podob shluků pro CNN

V případě logistické regrese a neuronových sítí jsou vstupy příslušných klasifikačních algoritmů přímo spočtené příznaky, resp. jejich vybraná podmnožina. CNN

vyžadují jako vstup čtvercový rastrový obraz. Tato kapitola popisuje přípravu rastrových vstupů do CNN užité v této práci, s cílem zachovat pro každý shluk informace o jeho tvaru a velikosti.

Pro zachování maximální velikostní věrohodnosti je nutné, převést všechny polygony shluků na rastr se stejným prostorovým rozlišením. Zároveň je ale potřeba splnit podmínku shodného rozlišení (ve smyslu počtu pixelů) pro všechny rastry vstupující do CNN. Pro splnění obou podmínek je nejprve nutné definovat jednu ze zmíněných proměnných. První možností je nejprve určit počet pixelů výsledných rastrů a prostorové rozlišení dopočítat na základě znalosti rozměrů největšího shluku budov na základě vzorce:

$$pr = \frac{\max(š_{max}, v_{max})}{r},$$

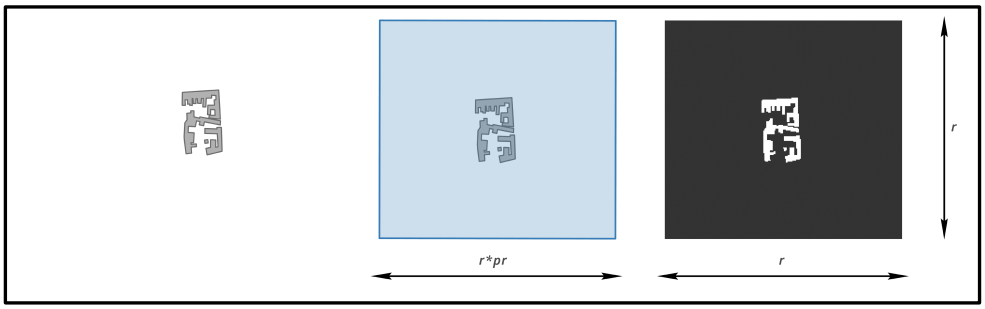
kde $š_{max}$,v_{max} jsou šířka a výška *bounding boxu* největšího shluku, r je počet pixelů výsledného rastru a *pr* hodnota hledaného prostorového rozlišení. Tento postup má však nevýhodu v nižší kontrole nad konečným prostorovým rozlišením, a tudíž i tvarovou věrohodností výsledných rastrových podob shluků. V krajním případě by prostorové rozlišení mohlo být tak malé, že by se jednotlivé budovy převedly pouze jako jeden pixel.

Druhou možností je nejprve určit prostorové rozlišení výsledných rastrů a počet pixelů následně dopočítat pomocí znalosti velikosti největšího shluku. Cílem je určit takovou hodnotu prostorového rozlišení, aby dokázala vystihnout tvary jednotlivých budov. Zároveň větší prostorové rozlišení zvyšuje počet pixelů ve výsledném rastru a tím i množství informace, které musí CNN zpracovat. Je proto je nutné najít kompromis mezi těmito požadavky. Po stanovení hodnoty prostorového rozlišení lze rozměry výsledného rastru určit pomocí znalosti velikosti největšího ze shluků. Rozlišení výsledného rastru lze spočítat jako:

$$r = (\max(š_{max}, v_{max}) \,//pr) + 1,$$

kde $š_{max}$,v_{max} jsou šířka a výška *bounding boxu* největšího shluku, pr je hodnota prostorového rozlišení a r je výsledná hodnota rozlišení, neboli počet pixelů v řádku i sloupci.

Zjednodušeně lze převod polygonových shluků do rastrové podoby rozdělit do tří kroků. V prvním kroku je vybrán vždy jeden shluk. Ve druhém kroku je okolo shluku vytvořen čtverec o straně *r*pr*. V posledním kroku jsou shluk a čtverec rasterizovány. V místech, kde se nachází polygon budov v příslušném shluku nabývá rastr hodnoty 1, jinde ve čtverci potom hodnoty 0. Schematicky je převod vektorové podoby shluku do rastru znázorněn na obrázku č. 32.



Obrázek 32: Proces rasterizace shluku budov
***Zdroj**: ČÚZK (2020d), vlastní zpracování*

Stanovení rozlišení výsledných rasterů pomocí největšího shluku je vhodné pokud nejsou ve velikosti shluků extrémní rozdíly. Výskyt pouze jednoho extrémně velkého shluku budov může vést k enormnímu nárůstu rozlišení výsledných rastrů. V případě velkých rozdílů ve velikosti shluků je možné, přistoupit k úpravě způsobu výpočtu výsledného rozlišení rastrů. Úprava spočívá v nahrazení největšího shluku takovým shlukem, jehož velikost bude větší než značná většina všech ostatních shluků, avšak nebude úplně největší. To kolik procent všech shluků bude menších následně odpovídá procentu shluků, které budou velikostně korektní, neboť u shluků větších než vybraný je nutné provést úpravu prostorového rozlišení. Pokud by k úpravě prostorové rozlišení nedošlo, tak by čtverec, který se vytváří okolo shluku budov a jeho velikost je definována pomocí prostorového rozlišení a velikosti vybraného shluku, neobsahoval všechny budovy shluku. Prostorové rozlišení pro shluky větší než vybraný shluk lze spočíst jako:

$$pr_i = \frac{\max(š_i, v_i)}{r},$$

kde $š_i, v_i$, jsou hodnoty šířky a výšky *bounding boxu* daného shluku, r je rozlišení výsledných rastrů.

Po analýze rozložení velikostí jednotlivých shluků, byla pro účely této práce, zvolena druhá možnost tvorby rastrových podob shluků, kdy je nejprve určeno prostorové rozlišení a až následně počet pixelů, včetně navržené úpravy výpočtu výsledného počtu pixelů pomocí nahrazení největšího shluku budov menším shlukem.

Diskuse volby prostorového rozlišení a velikosti shluku, jehož rozměry budou použity pro určení výsledného rozlišení, je uvedena v následující kapitole 4.6.

4.6 Implementace segmentačního algoritmu a příprava dat

Cílem této kapitoly je popsat implementaci algoritmů pro segmentaci a následnou tvorbu dat pro jednotlivé klasifikátory. Jak již bylo zmíněno v kapitole 4.1, pro automatizaci tvorby bylo využito programovacího jazyka *Python 3*. Sepsání algoritmů do podoby skriptů umožnilo opakovaně provádět dané výpočty.

Pro účely této diplomové práce byla zpracována data v rozsahu čtyř krajů v Česku. Počet budov v těchto krajích představuje více než 1/3 všech budov v Česku. Takovýto rozsah představuje dostatečné množství výsledných shluků budov jak pro výběr trénovací množiny, tak pro následnou analýzu přesnosti klasifikace jednotlivých klasifikátorů. Konkrétní počty budov a výsledných shluků v jednotlivých krajích jsou uvedeny v tabulce č. 3.

Kraj	Počet budov	% Počet budov v Česku	Počet shluků
Královehradecký kraj	204 478	6,8	37 517
Liberecký kraj	141 232	4,7	32 473
Pardubický kraj	184 962	6,2	32 261
Středočeský kraj	566 862	19,0	69 444
celkem	1 097 534	36,7	171 695

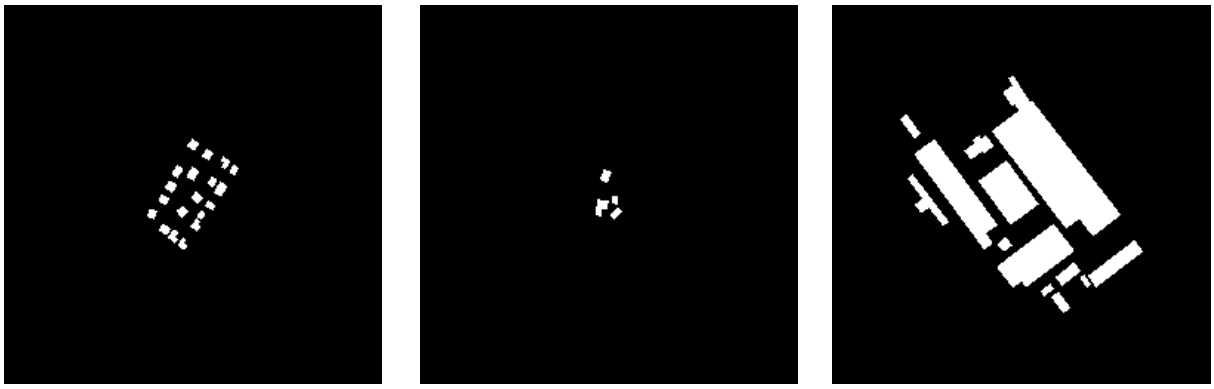
Tabulka 3: Počet budov a shluků
***Zdroj**: vlastní zpracování*

Pro snížení výpočetní náročnosti byla vstupní data rozdělena na čtyři části právě pomocí hranic zmíněných krajů. Rozdělení dat může mít negativní dopad na tvorbu shluků na hranici krajů. V celkovém množství shluků se ovšem jedná o zanedbatelné množství a toto nebylo dále řešeno.

Před spuštěním procesu segmentace bylo nutné určit proměnnou *d*, která definuje velikost bufferu vytvořeného okolo každé budovy (viz kapitola 4.3.2). Na základě experimentů se jako nejvhodnější ukázala hodnota 20 metrů. Čím je hodnota vyšší, tím je větší i vliv cestní sítě pro segmentaci budov do shluků a naopak. Veškeré výpočty týkající se procesu segmentace byly naprogramovány s využitím knihovny *ArcPy*. Proces segmentace je časově velice náročný. Pro každý z prvních tří krajů v tabulce č. 3 byla doba

běhu segmentačního algoritmu okolo 20 hodin. Pro Středočeský kraj přibližně dvojnásobek.

Výpočet charakteristik jednotlivých shluků budov přímo navazuje na segmentaci. Většina výpočtů charakteristik byla opět naprogramována s využitím knihovny *ArcPy*. Výjimku tvoří kategorie vzdálenostních charakteristik z kapitoly 4.4.3. Pro výpočet těchto charakteristik bylo využito funkcí, které poskytuje databázová nadstavba *PostGIS*. Funkce pro výpočet vzdáleností je nutné zadávat jako *SQL* dotazy. Pro možnost volání *SQL* dotazů v rámci skriptu byla využita již zmíněná knihovna *Psycopg*. Pro výpočet vzdálenostních charakteristik s využitím *PostGIS* bylo nutné požadovaná data nejprve nahrát v rámci běhu skriptu do databáze a následně provést dané výpočty. I přes tuto komplikaci, byl výpočet vzdálenostních charakteristik několikanásobně urychlen oproti funkcím, které poskytuje knihovna *ArcPy*. Doba výpočtu charakteristik je oproti segmentaci velice rychlá, v řádu několika minut. Výjimku tvoří výpočet průměrné výšky budov. Vzhledem k tomu, že byla jako zdroj výškových dat zvolena online služba, je nutné počítat s určitými omezeními. DMR i DMP lze ze zmíněného zdroje stahovat po dlaždicích o maximální velikosti 4048x4048 m. Právě stahování těchto dlaždic je velice pomalé, a ne vždy dojde ke správnému stažení a je nutné stahování opakovat. Z tohoto důvodu je proces výpočtu výšky budov přibližně stejně časové náročný jako proces segmentace.



Obrázek 33: Příklady rastrových podob shluků budov pro CNN
***Zdroj**: ČÚZK (2020d), vlastní zpracování*

Po doběhnutí procesu segmentace a výpočtu charakteristik je výsledek představován původní polygonovou vrstvou budov, ke které je připojena atributová tabulka. Atributová tabulka obsahuje jeden záznam pro každý shluk. Ve sloupcích jsou poté kromě identifikátoru hodnoty jednotlivých charakteristik. Po doběhnutí procesu pro všechny 4 kraje, byly dané 4 polygonové vrstvy sloučeny do jedné. Následně jsou přepočítány identifikátory jednotlivých shluků, aby nenastala situace, že jeden identifikátor bude odkazovat na více shluků.

Před vytvořením rastrových podob shluků je nutné stanovit dvě proměnné, viz kapitola 4.5. První proměnná *pr* definuje prostorové rozlišení. Na základě hledání kompromisu mezi tvarovou věrohodností rastrových podob shluků a celkovou datovou velikostí byla zvolena hodnota 2 metry. Následně je nutné stanovit výsledné rozlišení rastrů *r*. Na základě prozkoumání rozložení velikostí jednotlivých shluků bylo rozhodnuto, že výsledné rozlišení rastru bude určeno pomocí velikosti shluku, jehož percentil velikosti je roven 99,5 %. To znamená, že pouze u 0,5 % všech shluků nebudou výsledné rastry absolutně velikostně korektní. V tomto konkrétním případě je tato hodnota rovna 512 metrům. Výsledné rozlišení je poté, při hodnotě prostorové rozlišení 2 m, rovno 256x256 pixelů. Pro převod polygonů do rastrové podoby byla opět využita knihovna *ArcPy*. Výsledné rastry jsou ukládány do souboru *Cluster_ID.png*, kde *Cluster_ID* je již zmíněný unikátní identifikátor každého shluku vzniklý při segmentaci. Tvorba rastrových podob shluků je opět velice časově náročná. Převést veškeré shluky na rastr trvalo přibližně 60 hodin. Několik příkladů výsledných rastrů lze vidět na obrázku č. 33.

4.7 Klasifikace

Cílem této kapitoly je popsat průběh klasifikace zástavby pomocí tří různých klasifikátorů včetně jejich implementace. Nejdříve je v rámci této kapitoly popsán výběr trénovací a testovací množiny shluků budov. Následně je představen způsob zpracování dat před vstupem do klasifikátorů. Před klasifikací jsou představeny důvody pro výběr konkrétních klasifikátorů. Samotný průběh klasifikace pomocí jednotlivých klasifikátorů lze poté rozdělit do několika bodů:

1. příprava vstupních dat pro daný klasifikátor,
2. optimalizace klasifikátoru,
3. testování klasifikátorů.

Výběr trénovací množiny je nutným krokem před každou řízenou klasifikací (*supervised classification*). Cílem výběru trénovací množiny je nalézt několik typických představitelů předem definovaných tříd. Tito vzoroví zástupci následně slouží pro naučení daného klasifikátoru. Minimální doporučený počet zástupců každé třídy v trénovací množině je roven $N + 1$, kde N je počet příznaků vstupujících do klasifikace (Dobrovolný 1998). Pro účely této práce bylo ručně vybráno přibližně 90 zástupců pro každý typ zástavby. Toto číslo představuje přibližně čtyřnásobek počtu charakteristik jednotlivých shluků. Představitelé jednotlivých tříd byli vybíráni tak, aby co nejlépe

odpovídali vlastnostem jednotlivých typů zástavby, tak jak byly popsány v kapitole 3. Pro výběr zástupců bylo kromě vizuálního zhodnocení vektorové podoby shluku budov využito ortofoto snímků a informací o typu využití budov pocházejících z dat ZABAGED. Část této trénovací množiny bude též využita pro ověření přesnosti jednotlivých klasifikací.

Velice důležitou částí procesu klasifikace je úprava vstupních dat. Vstupní data velice často obsahují chybějící hodnoty, veliké odchylky nebo například kategoriální data (Huang, Li, Xie 2015). Cílem úpravy dat je zefektivnění fungování jednotlivých klasifikátorů (Alasadi, Bhaya 2017). Pro úpravu vstupních dat (charakteristiky shluků) pro účely této práce byly provedeny následující kroky:

1. **Odstranění nedefinovaných hodnot**

V případě, že se ve vstupních datech objeví nulové hodnoty, existují dvě možnosti, jak se s daným problémem vypořádat. První možností je dané záznamy smazat. Tím však přicházíme o určité množství dat. Druhou možností je nahradit nulové hodnoty jinou hodnotou, kterou může být průměr, medián či jiná konstanta. V této diplomové práci se nulové hodnoty objevují při výpočtu vzdáleností k nejbližším budovám v případě, že ve shluku není dostatečný počet budov pro výpočet dané charakteristiky. Pro výpočet *průměrné vzdálenosti k nejbližší budově* je nutné, aby byl shluk složen alespoň ze dvou budov. Pro výpočet *průměrné vzdálenosti ke třem nejbližším budovám* musí shluk obsahovat alespoň 4 budovy. Nahrazení nulových hodnot těchto dvou charakteristik pomocí průměru či mediánu není vhodné. Proto bylo rozhodnuto o nahrazení konstantou rovnou -1.

2. **Odstranění odchylek**

Cílem tohoto kroku je úprava takových hodnot, které se nacházejí mimo smysluplný rozsah hodnot vyplývající z definic daných charakteristik. V této práci se jedná o průměrnou výšku budov. Výška budovy nemůže logicky být nižší než 0 metrů a je též velice nepravděpodobné, aby průměrná výška shluků budov byla nižší než 3 metry. Proto byla u shluků s průměrnou výškou budov nižší než 3 m, upravena výška právě na hodnotu 3 metry.

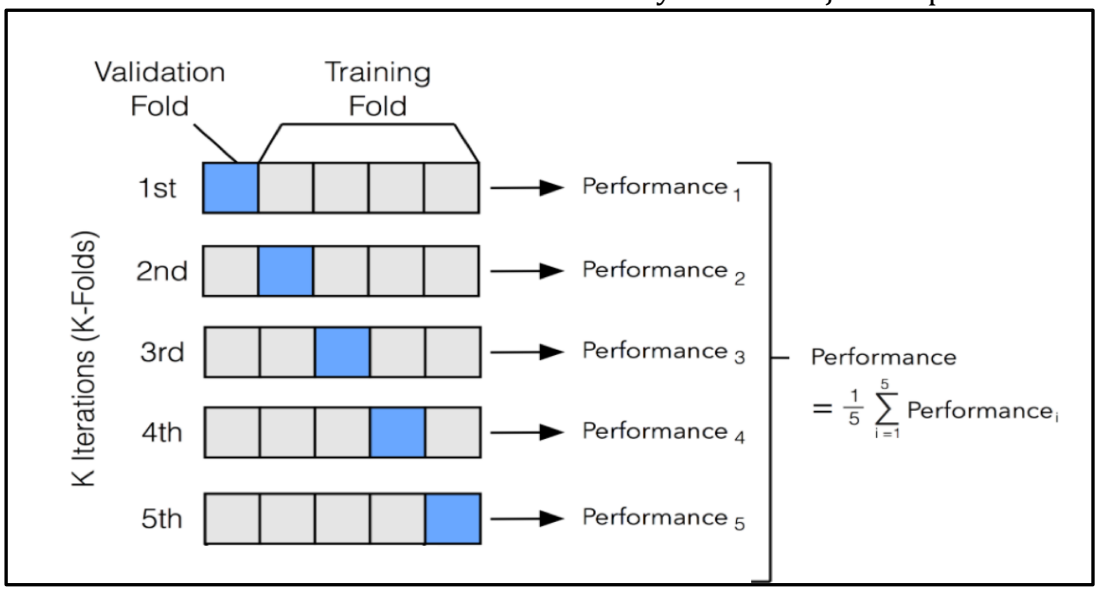
3. **Normalizace dat**

Při normalizaci dat dochází ke změně tvaru rozložení dat. Na rozdíl od škálování dat, kdy dochází pouze ke změně intervalu hodnot. Cílem tohoto kroku je splnění

předpokladu některých klasifikátorů, aby vstupní data měla normální rozdělení. Výslednou hodnotu charakteristiky pro daný shluk *i* lze po normalizaci spočíst jako:

$$z_i = \frac{x_i - \bar{x}}{s},$$

kde x_i je hodnota dané charakteristiky shluku, $\bar{x}$ je průměr dané charakteristiky přes všechny shluky a s je hodnota směrodatné odchylky spočtená z hodnot dané charakteristiky přes všechny shluky. Rozložení vzniklé touto normalizací se též nazývá *z-score*. Takové rozložení má směrodatnou odchylku rovnou jedné a průměr rovný nule.



Obrázek 34: Cross-validation
***Zdroj**: Liu (2019)*

Jak již bylo zmíněno v úvodu diplomové práce, dílčími cíli této práce je porovnat klasifikátory založené na popisných charakteristikách a vizuálním vjemu a též porovnat klasifikátory založené na tradičních statistických přístupech s klasifikátory využívajících neuronových sítí. Z těchto důvodů byly vybrány pro účely této diplomové práce celkem tři klasifikátory. Pro klasifikaci založenou na popisných charakteristikách jsou využity dva klasifikátory. Jako zástupce tradičních statistických přístupů byla vybrána skupina klasifikátorů založená na metodách strojového učení. Druhý klasifikátor pro klasifikace pomocí popisných charakteristik je založen na neuronových sítí. Pro klasifikaci na základě vizuálního vjemu byl vybrán pouze jediný klasifikátor, který využívá CNN.

Klasifikátorů spadajících do skupiny strojového učení je však velké množství. Z tohoto důvodu bylo na základě literatury vybráno sedm nejpoužívanějších klasifikátorů (Gredell et al. 2019), (Munkhdalai et al. 2019) (tabulka 3). Pro výběr nejvhodnějšího klasifikátoru byl pro každý z nich spuštěn test křížové validace (dále *cross-validation* test). Při tomto testu je trénovací množina rozdělena na *K* stejně početných částí. Pro

jednotlivé části platí, že rozložení počtu prvků v jednotlivých třídách odpovídá rozložení v celé množině dat. Při každé z *K* iterací je vždy jedna část označena za testovací a zbylé za trénovací. Následně je daný klasifikátor naučen pomocí dané trénovací množiny. Testovací množina poté slouží k výpočtu přesnosti klasifikace. Přesnost klasifikace je v tomto případě definována jako poměr počtu správně klasifikovaných shluků ku počtu všech testovacím. V následující iteraci je jako testovací množina zvolena jiná část původních dat, tak jak to lze vidět na obrázku č. 34. Následně jsou přesnosti klasifikace při jednotlivých iterací zprůměrovány. Pro účely této diplomové práce byla zvolena hodnota $K = 5$. Pro početnější trénovací množiny je možné volit větší hodnotu *K*. Jednotlivé klasifikátory byly použity v základním nastavení, tak jak jsou implementovány v knihovně *Scikit-learn*. Jako vstupní data bylo použito všech 22 charakteristik daných shluků. Výsledky testů lze vidět v tabulce č. 4. Jako nejvhodnější kandidáti se dle výsledků ukázaly klasifikátory *Gradient Boost* a *Logistická regrese*. Oba tyto klasifikátory ukázaly velice dobré předpoklady pro co nejpřesnější klasifikaci zástavby. Vzhledem k tomu, že *Logistická regrese* dosáhla nepatrně lepšího výsledku, byl zvolen právě tento klasifikátor jako zástupce metod strojového učení pro klasifikaci zástavby.

Klasifikátor	Přesnost
K –nejbližších sousedů	0,8147
Support Vector Machine	0,8148
Logistická regrese	0,8782
Rozhodovací strom	0,8187
Maximum Likelihood	0,7999
Random Forest	0,8484
Gradient Boost	0,8634

Tabulka 4: Výběr klasifikátoru
***Zdroj**: vlastní zpracování*

4.7.1 Logistická regrese

Cílem této kapitoly je popsat implementaci metod, které jsou použity pro zvýšení klasifikační přesnosti s využitím logistické regrese. Jako první je představen způsob implementace metod pro výběr nejvhodnějších charakteristik shluků. Následně je popsán proces ladění parametrů logistické regrese. Na závěr jsou analyzovány výsledky

tohoto klasifikátoru. Srovnání výsledků dosažených pomocí *Logistické regrese* a zbývajících klasifikátorů založených na neuronových sítí lze nalézt v kapitole č. 5.

4.7.1.1 Implementace výběru nejdůležitějších parametrů

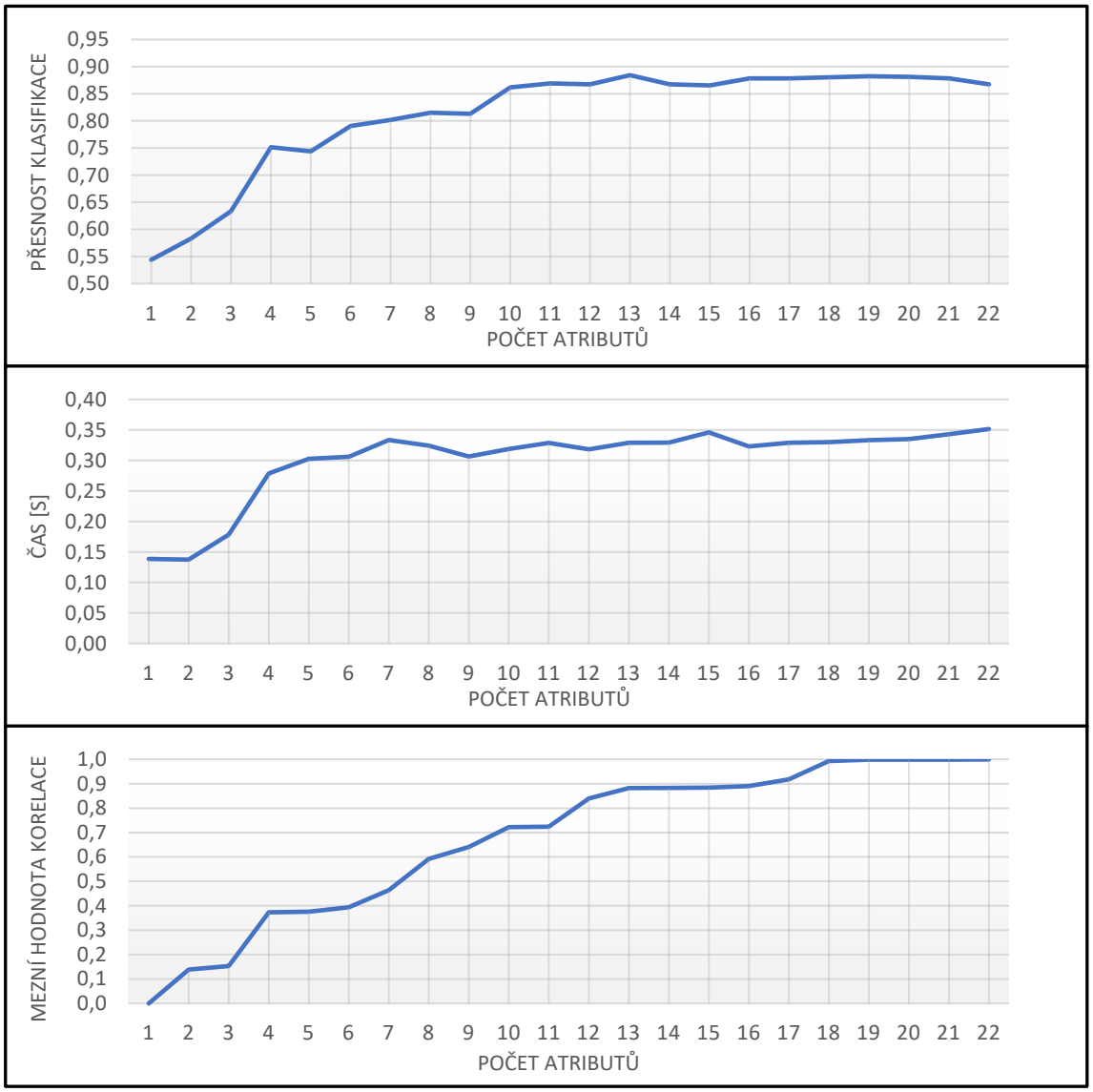
Důvody pro využití metod pro výběr nejvhodnějších atributů byly již popsány v kapitole č. 2.6. Cílem této kapitoly je popsat způsob implementace jednotlivých metod pro výběr atributů vstupujících do procesu klasifikace, včetně jejich výsledků a vybrat nejvhodnější metodu pro použití dále při vlastní klasifikaci.

1. **Výběr na základě korelace**

   Při této metodě dochází k odstraňování atributů, jejichž vzájemná korelace je vyšší než daná mez. Bližší popis metody lze nalézt v kapitole 2.6. Pro určení, při jaké míře korelace dochází k nejvyšší přesnosti klasifikace byl využit jednoduchý *for* cyklus. Uvnitř tohoto cyklu dochází k postupnému navyšování mezní hodnoty korelačního koeficientu. Tato hodnota poté vstupuje do funkce pro výběr atributů na základě korelačního koeficientu, tak jak byla popsána v kapitole 2.4. Poté, co daná funkce vybere atributy odpovídající danému kritériu, je proveden dále *cross-validation* test na těchto vybraných charakteristikách. Po proběhnutí tohoto testu je zaznamenána výsledná hodnota přesnosti klasifikace při dané míře korelačního koeficientu. Kromě přesnosti klasifikace byl pro každý *cross-validation* test měřen i čas běhu každého *cross-validation* testu. Výsledky této metody výběru atributů lze vidět na grafu č. 1.

   Z grafů č. 1 lze vyčíst, že nejvyšší přesnosti klasifikace je dosaženo při využití pouze 13 atributů, což odpovídá odstranění atributů s vyšší hodnotou korelace než přibližně 0,88. Přesnost klasifikace dosahuje v tomto případě 88,5 %. Avšak již od využití 10 atributů nedochází k velkému nárůstu přesnosti klasifikace. Časová náročnost provedení *cross-validation* testu se rychle zvyšuje do využití 7 atributů, poté je již časová náročnost stejná pro jakýkoliv počet využitých charakteristik. Vzhledem k tomu, že rozdíl v časové náročnosti mezi využitím 10 a 13 charakteristik je zanedbatelný, bylo by vhodné při využití této metody ponechat 13 atributů, neboť poskytují nepatrně vyšší

klasifikační přesnost. Jinak řečeno, při využití této metody by bylo vhodné nastavit mezní hodnotu korelace na hodnotu 0,8826.

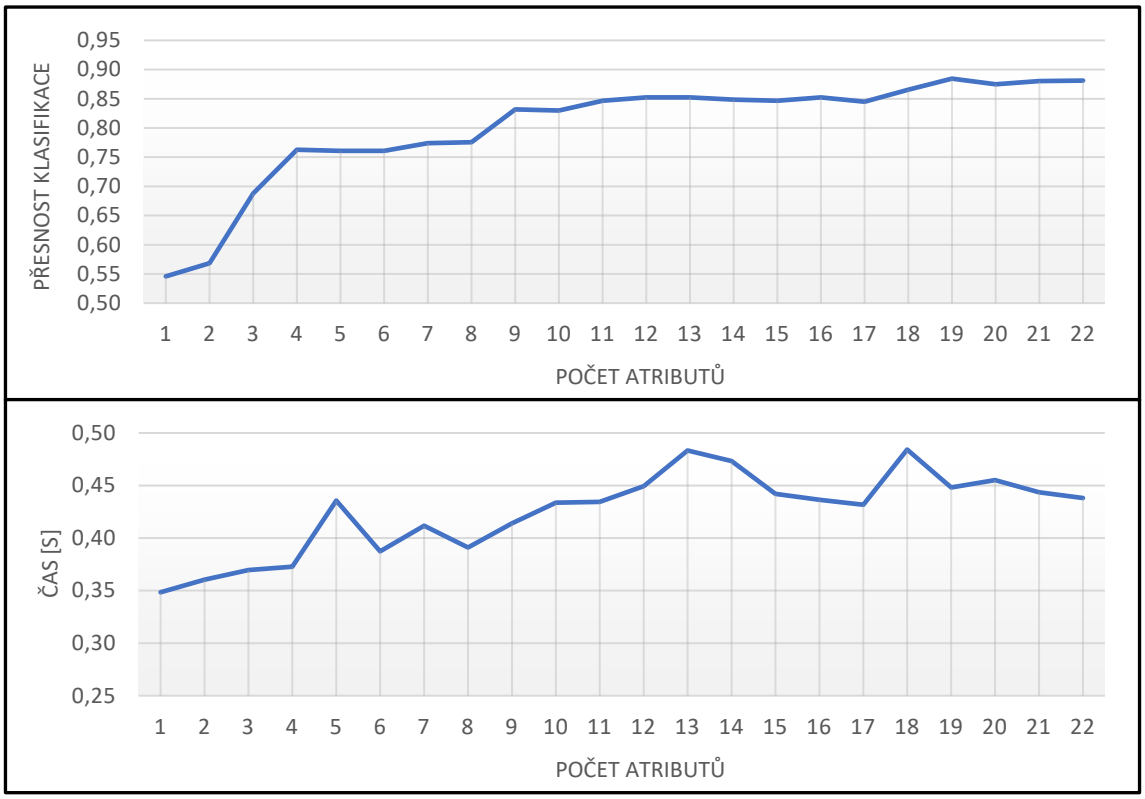


Graf 1: Výběr atributů pro LR na základě korelace
***Zdroj:** vlastní zpracování*

2. Výběr atributů pomocí ANOVA testu

Pro implementaci této metody byla využita funkce, z již zmíněné knihovny *Scikit-Learn*. Bližší popis fungování metody lze nalézt v kapitole 2.6. Pro zjištění nejvhodnějších atributů byl opět využit *for* cyklus. Na rozdíl od předchozí, vstupuje do této funkce počet nejvhodnějších atributů. Proto uvnitř cyklu dochází pouze postupnému nárůstu počtu výsledných atributů. Stejně jako u předchozí metody, byl pro ověření přesnosti klasifikace využit *cross-validation* test a měřen čas. Výsledky této metody lze vidět na grafu č. 2.

Z grafu závislosti přesnosti klasifikace na počtu atributů lze vyčíst podobné závěry jako u předchozí metody. Obdobně jako u předchozí metody lze s přibývajícím počtem atributů pozorovat zvyšující se úspěšnost klasifikace. Maximální úspěšnost byla v tomto případě dosažena při 19 zachovaných atributech. Úspěšnost klasifikace v tomto případě dosahuje 88,4 %. Avšak již od počtu 12 atributů nedochází k výraznému nárůstu přesnosti klasifikace. Graf závislosti času na počtu atributů již není tak vypovídající jako u předchozí metody, přesto lze říct, že s rostoucím počtem atributů, roste i čas nutný pro klasifikaci.

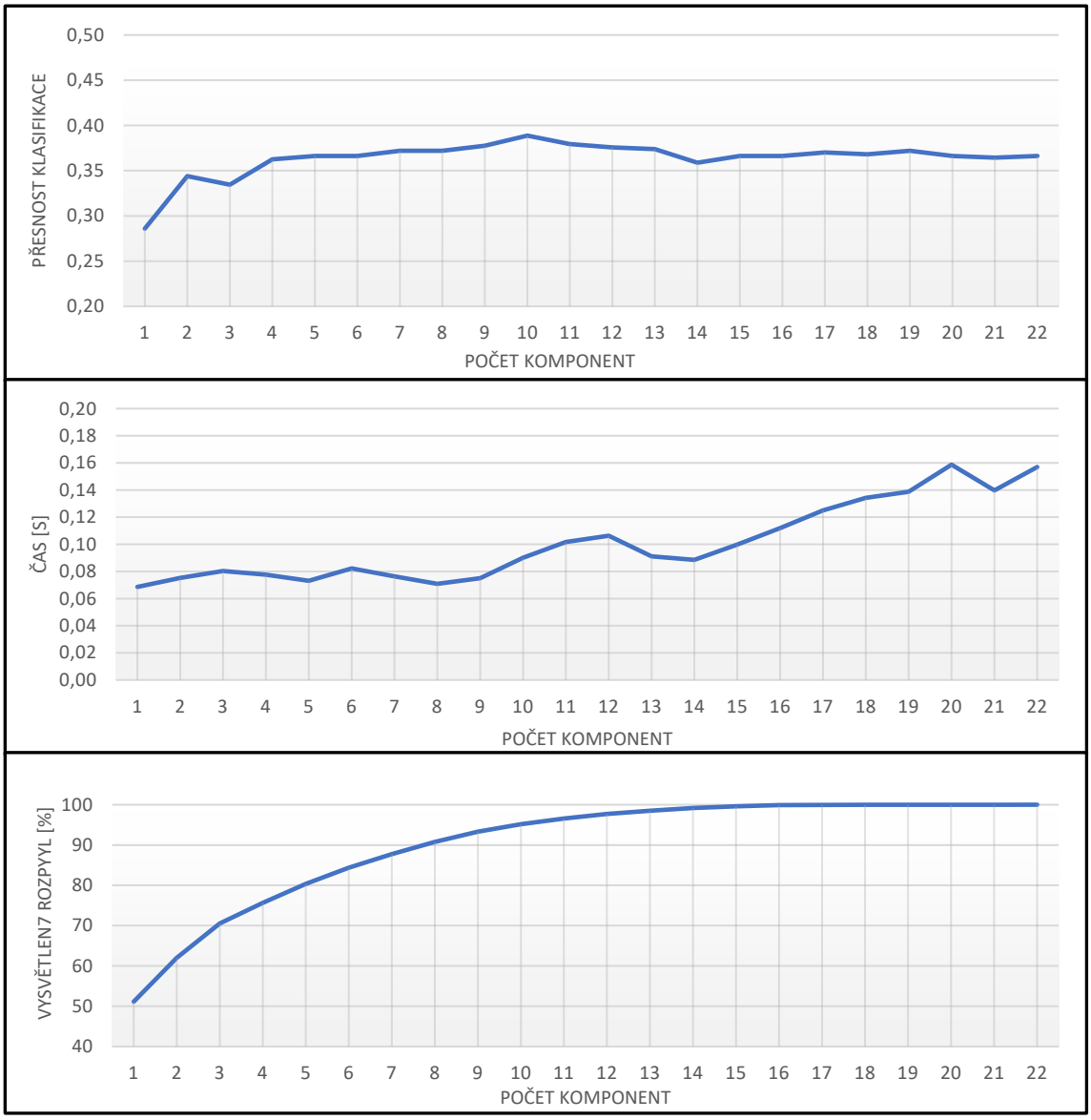


Graf 2: Výběr atributů pro LR na ANOVA testu
***Zdroj:** vlastní zpracování*

3. Výběr atributů pomocí metody PCA

Jak již bylo zmíněno v kapitole 2.6, při této metodě nedochází k přímému výběru nejdůležitějších atributů, ale dochází k tvorbě nových. Pro implementaci této metody byla opět využita funkce z knihovny *Scikit-Learn*. Pro zjištění ideálního počtu nově vzniklých komponent (charakteristik) byl opět využit *for* cyklus. V tomto případě dochází uvnitř cyklu k navyšování počtu výsledných komponent. Pro zjištění výsledné přesnosti klasifikace byl opět využit *cross-validation* test. Kromě přesnosti a času bylo zaznamenáváno i celkové procento rozptylu vysvětlené pomocí nově vzniklých komponent (graf 3). Již na první pohled je zřejmé, že metoda PCA není vhodná pro použití v kombinaci s *Logistickou regresí*. Výsledná přesnost klasifikace pomocí nově

odvozených charakteristik je velice nízká. Počet výsledných komponent tento jev nijak neovlivňuje, i přesto že procenta vysvětlené variability logaritmicky rostou.



Graf 3: Výběr atributů pro LR na základě korelace
***Zdroj:** vlastní zpracování*

Ze všech tří zkoumaných metod pro výběr nejpřínosnějších charakteristik se jako vhodné ukázaly první dvě. Jejich přínos pro zvýšení přesnosti klasifikace však není příliš velký. Oproti použití všech atributů se jedná o zlepšení pouze v řádu několika desetin procent. Větší přínos těchto metod lze očekávat při snížení celkového množství dat vstupujících do klasifikátoru a tím urychlení procesu klasifikace. Vzhledem k tomu, že metoda pro výběr nejdůležitějších atributů pomocí korelace dosáhla maximální hodnoty úspěšnosti při nižším počtu atributů, byla vybrána právě tato metoda jako nejvhodnější pro další použití. V tabulce č. 5 lze vidět 13 vybraných nejpřínosnějších atributů při využití mezní hodnoty korelace 0,8826.

Vybrané charakteristiky			
Součet plochy budov	Průměrná plocha budovy	Součet obvodů budov	Průměrný obvod budovy
Obvod konvexní obálky	Poměr plochy budov a konvexní obálky	Poměr šířky a délky opsaného obdélníku s minimální šířkou	Směrodatná odchylka ve směru
Průměrná vzdálenost k silnici	Průměrná vzdálenost k nejbližší budově	Průměrná vzdálenost ke třem nejbližším budovám	Průměrná výška budovy
Poměr plochy budov k ploše polygonu z první části segmentace			

Nevybrané charakteristiky		
Plocha konvexní obálky	Délka opsaného obdélníku s minimální plochou	Plocha opsaného obdélníku s minimální plochou
Délka opsaného obdélníku s minimální šířkou	Plocha opsaného obdélníku s minimální šířkou	Šířka opsaného obdélníku s minimální plochou
Šířka opsaného obdélníku s minimální šířkou	Směrodatná odchylka plochy budov	Počet budov

Tabulka 5: Seznam výsledných atributů pro LR
***Zdroj**: vlastní zpracování*

4.7.1.2 Ladění parametrů logistické regrese

Cílem tohoto kroku je pokusit se zvýšit přesnost klasifikace pomocí úpravy proměnných vstupujících do klasifikace pomocí logistické regrese. Na základě rešerše metod pro optimalizaci parametrů funkce (kapitola 2.5) a zkušeností s časovou náročností logistické regrese při hledání optimálních atributů byl vybrán algoritmus *GridSearch*.

Jak pro implementaci algoritmu *GridSearch*, tak pro implementaci logistické regrese byla opět využita knihovna *Scikit-Learn*. Implementace *Logistické regrese* v této knihovně umožňuje upravit více než 10 různých vstupních parametrů. Na základě rešerše obecného fungování logistické regrese (kapitola 2.2.1) a doporučení obsažených v dokumentaci této knihovny (Scikit-Learn 2018) byly vybrány 4 parametry. První

parametr (*Solver*) definuje výběr algoritmu, který obstarává iterační proces fitování logistické křivky (obrázek 6) na vstupní data. Konkrétní algoritmy včetně jejich popisu a odkazu na literaturu lze opět nalézt v dokumentaci knihovny. Druhý parametr (*C*) určuje míru regularizace. Smyslem regularizace je přimět model lépe reagovat na odlehlé hodnoty ve vstupních datech. Tato schopnost umožní modelu lépe generalizovat, neboli zvýšit přesnost klasifikace při klasifikaci neznámých dat (Piatetsky-Shapiro 2020).Jak již bylo zmíněno v kapitole 2.2.1, fitování logistické křivky je iterační proces. Třetí zkoumaný parametr (*Max_iter*) určuje právě maximální počet iterací. Poslední parametr (*Toll*) doplňuje parametr předcházející. Jeho cílem je definovat prahovou hodnotu, kdy je logistická křivka již nafitovaná s určitou přesností. Pokud je této prahové hodnoty dosaženo, iterační proces je ukončen. V případě, že této hodnoty není dosaženo, je iterační proces ukončen při dosažení maximálního počtu iterací definovaného předchozím parametrem.

Rozsah hodnot, kterých můžou jednotlivé parametry nabývat je opět definován v dokumentaci klasifikátoru. Algoritmus *GridSearch* vyžaduje na vstupu předem definovaný seznam hodnot, kterých může daná proměnná nabývat. Z tohoto důvodu je nutné, v případě numerických proměnných, kromě rozsahu určit též počet a rozložení hodnot v přípustném intervalu.

Pro vytvoření seznamu hodnot se nejčastěji využívají generátory. Tyto funkce vytvoří seznam čísel mezi počáteční a koncovou hodnotou s požadovaným rozložením. Základní rozložení je rovnoměrné. Při tomto rozložení jsou rozestupy mezi každými dvěma čísly stejné. Mezi další rozložení patří normální nebo logaritmické. Avšak vzhledem k tomu, že je zkoumána každá kombinace hodnot parametrů, nemusí být jednotlivé hodnoty v seznamu seřazené.

Při stanovování počtu hodnot jednotlivých parametrů je nezbytné si uvědomit, že díky zákonům kombinatoriky dochází k exponenciálnímu navyšování počtu všech zkoumaných kombinací. Z tohoto důvodu je vhodné provést hledání ideálních hodnot parametrů ve více kolech. Při prvním kole prohledávání jsou nastaveny široké rozsahy, kterých mohou hodnoty jednotlivých parametrů nabývat. V závislosti na výsledcích prvního kola je možné v dalším kole rozsahy zúžit. To při zachování stejného počtu prvků v obou kolech hledání umožní, najít nejvhodnější hodnoty jednotlivých proměnných v mnohem kratším čase. Při snižování rozsahu jednotlivých parametrů je nutné být velice

opatrný, neboť při chybném zúžení již nemusí dojít k nalezení nejideálnějších hodnot parametrů.

Pro nalezení ideálních hodnot parametrů byl pro každou kombinaci vstupních hodnot parametrů proveden *cross-validation* test. Kombinace s nejvyšší přesností klasifikace je poté označena jako nejvhodnější. V tabulce č. 6 lze vidět počáteční rozsahy hodnot, pro jednotlivé parametry *Logistické regrese*, hledaných pro účely této diplomové práce. Rozložení hodnot v těchto intervalech není uvedeno z důvodu, že hledání ideálních parametrů proběhlo ve více kolech. Ve druhém sloupci tabulky je poté nejlepší nalezená kombinace. Celkově bylo prohledáno cca 1 000 000 kombinací. Časová náročnost takového počtu kombinací byla v prostředí *Google Colab* v řádu několika hodin. Zhodnocení klasifikační přesnosti po provedení ladění parametrů lze nalézt v následující kapitole.

parametr	rozsah	ideál
Solver	'newton-cg','lbfgs', 'saga'	'newton-cg'
C	<0,1;10000>	12,54
Toll	<0,00001;10000>	0,42
Max_iter	<10;10000>	120

Tabulka 6: Grid Search
***Zdroj**: vlastní zpracování*

4.7.1.3 Analýza výsledků klasifikace pomocí logistické regrese

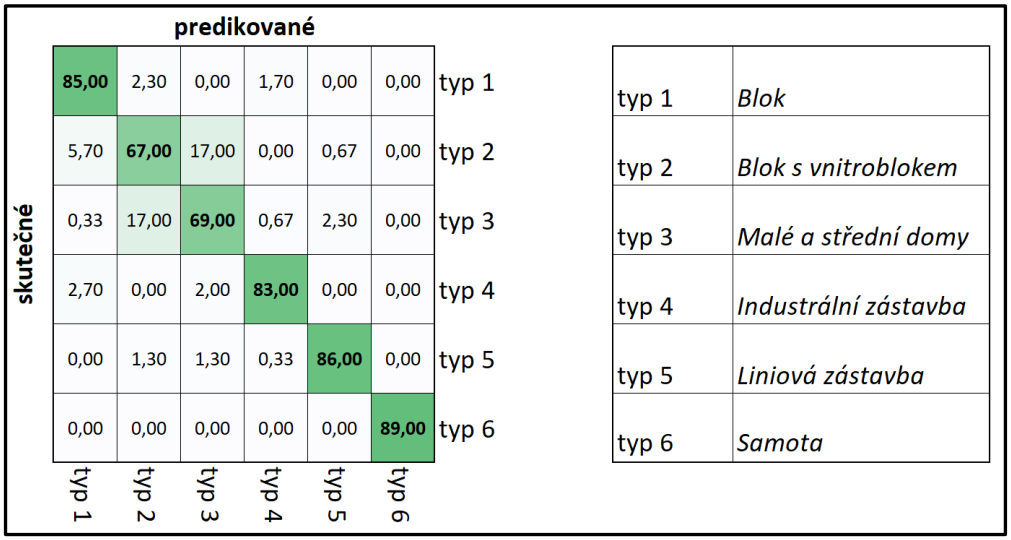
Tato kapitola práce je věnována bližšímu pohledu na výsledky klasifikace zástavby pomocí logistické regrese. Pro hodnocení výkonnosti klasifikátoru logistické regrese byla zvolena chybová matice, respektive metriky z ní vycházející. Popis jednotlivých metrik lze nalézt v kapitole 2.7.

Hodnoty v chybové matici lze získat pomocí několika různých přístupů. První možností je využít všechna trénovací data zároveň jako testovací. Výhodou tohoto postupu je fakt, že klasifikátor je natrénován i otestován na maximálním možném počtu vstupních prvků. Tento přístup poskytuje informaci, jak dobře je klasifikátor klasifikuje známá data. Velkou nevýhodou je skutčenost, že pomocí tohoto přístupu nelze zjistit, jak bude klasifikátor reagovat na neznámá data. Druhou možností je proto rozdělit vstupní data na trénovací a testovací část. Tento přístup již lépe charakterizuje výkonnost klasifikátoru při použití na neznámých datech. Hodnoty chybové matice jsou však při tomto přístupu ovlivněny výběrem dat pro trénovací a testovací část.

Z těchto důvodů byl pro tuto diplomovou práci zvolen postup založený *na cross-validation* testu. Na rozdíl od *cross-validation* testu není při každé iteraci testu vypočtena přesnost klasifikace, ale je sestrojena chybová matice. Následně jsou matice z jednotlivých iterací *cross-validation* testu sečteny. Celý *cross-validation* test je pro získání reprezentativnějších výsledků 3x zopakován. Výsledná chybová matice je získána zprůměrováním chybových matic, ze tří opakování *cross-validation* testu.

Na obrázku č. 35 již lze vidět chybovou matici pro klasifikaci zástavby pomocí klasifikátoru logistické regrese. Jako vstup do klasifikátoru bylo využito 13 nejvhodnějších atributů vybraných v kapitole 4.7.1.1 metodou založenou na korelaci. Samotný klasifikátor byl poté nastavený pomocí hodnot parametrů vybraných pomocí algoritmu *GridSearch*. Vstupní data byla pro *cross-validation* test rozdělena na 5 částí (k=5). To znamená, že při každé iteraci byl klasifikátor natrénován na přibližně 427 shlucích budov a otestován na přibližně 107. Součet hodnot ve výsledné chybové matice (obrázek 35) poté odpovídá počtu prvků v ručně vybrané trénovací množině. Hodnoty metrik vycházejících z chybové matice jsou zaneseny v tabulce č. 7.

Jak lze vyčíst z tabulky č. 7, výsledná přesnost klasifikace dosáhla hodnoty téměř 90 %. Oproti klasifikaci zástavby se základním nastavením klasifikátoru logistické regrese se jedná přibližně o 1 % vyšší hodnotu a oproti výsledkům klasifikace při využití všech charakteristik se jedná přibližně o 2% zvýšení přesnosti klasifikace.



Obrázek 35: Chybová matice pro klasifikace zástavby pomocí LR
***Zdroj**: vlastní zpracování*

Pro určení výkonnosti klasifikátoru ve vztahu k jednotlivým třídám je nejlepších ukazatelem *F-score*, neboť pro tento klasifikační problém jsou hodnoty uživatelské i zpracovatelské přesnosti stejně důležité. Nejlépe ze všech typů zástavby byla

klasifikována třída *Samota*. Všechny shluky tohoto typy byly klasifikovány správně (hodnota *recall = 1*). A žádný ze shluků jiných typů zástavby nebyl označen jako *Samota* (hodnota *precision = 1*).

Typ zástavby	Precision	Recall	F-score
Blok	0,907	0,955	0,930
Blok s vnitroblokem	0,765	0,741	0,753
Malé a střední domy	0,773	0,773	0,773
Industriální zástavba	0,968	0,946	0,957
Liniová zástavba	0,967	0,967	0,967
Samota	1,000	1,000	1,000
		Accuracy	**0,897**

Tabulka 7: Hodnocení klasifikace zástavby pomocí LR
***Zdroj**: vlastní zpracování*

Velice dobře byly též klasifikovány třídy *Liniová* a *Industriální zástavba*. Hodnoty *F-score* je pro tyto typy zástavby rovna 0,97 respektive 0,96. Pouze v několika málo případech byly shluky těchto tříd označeny klasifikátorem jako některá jiná třída. Stále vysoká hodnota *F-score* byla naměřena u typu *Blok*. Tato třída byla v rámci klasifikace několikrát označena jako třída *Blok s vnitroblokem* a *Industriální zástavba*. Též v několika případech došlo k chybnému označení typu *Blok s vnitroblokem* jako typu *Blok*. Již o poznání hůře byly klasifikovány dva zbývající typy zástavby: *Malé a střední domy* a *Blok s vnitroblokem*. Hodnoty *F-score* těchto dvou tříd jsou rovny 0,78 respektive 0,75. Chyby při klasifikaci těchto dvou typů byly způsobeny zejména jejich záměnou. Přibližně každý pátý shluk těchto typů zástavby byl označen jako shluk náležící k druhému typu zástavby.

Vyrovnané hodnoty míry preciznosti a úplnosti, respektive vyrovnané počty v řádcích a sloupcích chybové matice pro jednotlivé třídy svědčí o tom, že žádná ze tříd nebyla výrazně při klasifikaci nadhodnocena či podhodnocena. Jinými slovy, rozložení počtů shluků v jednotlivých typech před klasifikací odpovídá rozložení počtů shluků v jednotlivých typech i po klasifikaci.

4.7.2 Neuronová síť

Cílem této kapitoly je popsat implementaci klasifikace zástavby pomocí neuronové sítě s využitím spočítaných příznaků. Nejdříve je popsán proces hledání optimální architektury neuronové sítě včetně procesu ladění hyperparametrů sítě. Následně je popsán proces hledání ideálních charakteristik shluků budov, včetně diskuse možného využití výsledků klasifikace pomocí logistické regrese pro rozšíření trénovací množiny.

Na závěr této kapitoly jsou analyzovány výsledky klasifikace zástavby pomocí neuronové sítě.

4.7.2.1 Architektura sítě a ladění hyperparametrů

Vzhledem k tomu, že cílem této diplomové práce je zejména ověřit, zda je vůbec možné klasifikovat zástavbu pomocí neuronových sítí, byla zvolena spíše jednodušší architektura sítě. Ve zvolené architektuře je využito bloků neboli skupin vrstev. Vrstvy v bloku mají pevně dané pořadí. Pro účely této práce bylo využito bloků skládajících se pouze ze tří vrstev. První vrstvu představuje *Dense layer*, na kterou přímo navazuje vrstva obsahující aktivační funkci. Poslední vrstvou bloku je *Dropout layer*. Tyto bloky jsou poté skládány za sebe. Počet těchto bloků představuje jednu z proměnných, která společně s dalšími hyperparametry vstupuje do algoritmu pro hledání optimálních hodnot daných proměnných. Každá neuronová síť musí být ukončena pomocí závěrečné vrstvy (*Output layer)*, která je představována *Dense layer* a aktivační funkcí. Jak již bylo zmíněno v kapitole 2.4, počet neuronů v závěrečné *Dense layer* je v případě klasifikační úlohy vždy roven počtu klasifikovaných tříd. V případě této diplomové práce je počet neuronů roven 6. Závěrečná aktivační funkce je pro klasifikační úlohy vždy *Softmax* (viz kapitole 2.4). Na rozdíl od závěrečné vrstvy, není počet neuronů první skryté vrstvy ovlivněn počtem tříd či počtem vstupních charakteristik. Z tohoto důvodu, není nutné před první blok vrstev vkládat dodatečné vrstvy. Symbolicky je architektura neuronové sítě při takto navrženém bloku znázorněna v tabulce č. 9.

Proměnná	Rozsah
Počet neuronů	[16,32,64,128,256,512,1024]
Velikost dropoutu	(0;1)
Počet bloků	(1,2,3, …,10)
Aktivační funkce	[ReLU,Sigmoid]
Optimalizátor	[SGD, RMSProp, Adam, Adagrad, Adamax]
Learning rate	(0;1)
Batch Size	[16,32,64,128]

Tabulka 8: Rozsah hyperparametrů pro neuronovou síť
***Zdroj**: vlastní zpracování*

Pro *Dense layer* je nutné určit minimálně jeden parametr – počet neuronů. Pro druhou vrstvu bloku je nutné zvolit konkrétní aktivační funkci (viz kapitole 2.4). *Dropout layer* je poté charakterizována jediným parametrem, a to právě hodnotou *dropoutu*. Počet

neuronů a výše *dropoutu* jsou proměnné, které mohou být pro jednotlivé bloky v neuronové síti rozdílné. Aktivační funkce je volena společně pro všechny *Dense layer* v celé síti. Další proměnou, která bude vstupovat do optimalizační funkce, je již zmíněný počet bloků použitých pro sestavení celé neuronové sítě. Velmi důležitá je též volba optimalizačního algoritmu (*Optimizer*), jehož úkolem je úprava vah uvnitř jednotlivých vrstev pro minimalizaci ztrátové funkce (*Loss Function*). Poslední dvě proměnné představuje velikost *Batch size* a *Learning Rate*. *Batch size* udává počet prvků trénovací množiny, které jsou využity pro jeden průchod neuronovou sítí v rámci jedné epochy. Vlastnosti proměnné *Learning Rate* již byly popsány v kapitole 2.4. Seznam a přípustné hodnoty všech proměnných, jejichž hodnoty byly určeny pomocí optimalizační funkce, jsou uvedeny v tabulce č. 8. Celkový počet proměnných je závislý na počtu zvolených bloků vrstev v neuronové síti. Konkrétně pro navrženou architekturu pro účely této práce je výsledný počet proměnných = $4 + k * 2$, kde k je počet použitých bloků pro sestavení sítě.

Vrstva	Parameter	
Dense Layer	128	Blok 1
Aktivační funkce	ReLU	
Dropout layer	0,3	
Dense layer	32	Blok 2
Aktivační funkce	ReLU	
Dropout layer	0,4	
Dense layer	6	Závěrečné vrstvy
Aktivační funkce	Softmax	

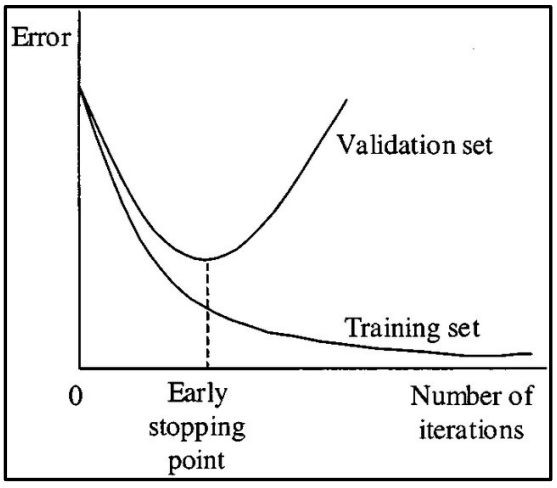
Optimalizátor	Adam
Learning rate	0,001
Batch size	16

Tabulka 9: Výsledná architektura neuronové sítě
***Zdroj**: vlastní zpracování*

Pro optimalizaci daných proměnných byla zvolena metoda založená na *Bayesovské optimalizaci*, která byla popsána v kapitole 2.6. Vzhledem k tomu, že naučení neuronové sítě a následná klasifikace trvá mnohonásobně déle než v případě klasifikátoru logistické regrese, byla upřednostněna tato metoda před metodou *GridSearch*. Pro implementaci této metody byla využita knihovna *Optuna*. Tato knihovna byla vybrána z důvodu dobré kompatibility s knihovnou *Tensorflow 2.0*, která byla využita pro implementaci neuronových sítí.

Pro zjištění vhodnosti dané kombinace byl opět použit *cross-validation* test. Pro použití s neuronovými sítěmi byl však lehce upraven. Na rozdíl od logistické regrese, není u neuronových sítí zaručeno, že při učení neuronové sítě na stejných vstupních datech dostaneme i stejné výsledné váhy uvnitř neuronové sítě. Jinými slovy, neuronová síť naučená a otestovaná na absolutně stejných datech, nemusí vykázat vždy stejné hodnoty přesnosti klasifikace. Tento fakt spočívá v několikanásobném využití náhodné veličiny uvnitř neuronové sítě. Jedná se zejména o náhodné generování výchozích vah *Dense layers*. Prvek náhodnosti je též obsažen i v *Dropout layers*, kdy jsou náhodně vybírány neurony, jejichž výstupy nebudou využity. Rozdílné výsledky lze též získat při využití CPU(*central processing unit*) nebo GPU pro výpočet (Brownlee 2019).

První úprava *cross-validation* testu spočívá v rozdělení testovací části na validační a testovací část (obrázek 39). Obě tyto části obsahují stejný počet shluků budov a zároveň je shodný i počet shluků budov náležící jednotlivým typům zástavby. Účelem validační části dat je během procesu učení průběžně ověřovat, zda nedochází k přeučení sítě (viz. kapitola 2.4). Testovací část slouží stejně jako v případě logistické regrese pro získání hodnoty přesnosti klasifikace jako poměru správně klasifikovaných shluků budov ku všem klasifikovaným shlukům budov. Náhodné rozdělení testovací části na dvě zmíněné části je prováděno před každým učením neuronové sítě.



Obrázek 36: Early Stopping
***Zdroj**:* Ren, Bai (2011)

Druhá úprava *cross-validation* testu spočívá v opakovaném provedení jednotlivých iterací. Pro účely této práce byla každá iterace provedena 5x. Jinými slovy, na stejných vstupních (trénovacích) datech je neuronová síť natrénována několikrát. Po naučení neuronové sítě byla vždy spočtena přesnost klasifikace pomocí testovací části dat. Pro

každou iteraci *cross-validation* testu byly tyto výsledky zprůměrovány. Po doběhnutí *celého cross-validation* testu byly zprůměrovány i výsledky jednotlivých iterací.

Pro částečné snížení časové náročnosti bylo využito funkce (*EarlyStopping*) z knihovny *Tensorflow*, která zastaví proces učení neuronové sítě v případě, že již nedochází ke zlepšení některé z metrik počítaných při každé epoše. Nejčastěji se pro tyto účely využívá hodnota *Validation Loss*. Jedná se o hodnotu ztrátové funkce počítané z validačních dat po dokončení každé epochy. Výhodou této funkce je též skutečnost, že dokáže předcházet přeučení sítě. Projevem přeučení může být právě nárůst hodnoty *Loss function* (obrázek 36). Při použití této funkce je vhodné nastavit vyšší hodnotu počtu epoch, které by proběhly v případě, že by nedošlo k zastavení učení pomocí funkce *EarlyStopping*. Pro účely této práce byl maximální možný počet epoch nastaven na 200.

Celkově bylo otestováno více než 10 000 různých kombinací vstupních hodnot jednotlivých proměnných. Jako vstupní data byla použita ručně vybraná trénovací množina čítající přibližně 500 shluků budov. Celková doba běhu optimalizačního algoritmu pro tento počet kombinací byla okolo 60 hodin. Výslednou architekturu sítě včetně hodnot jednotlivých hyperparametrů, lze vidět v tabulce č. 9.

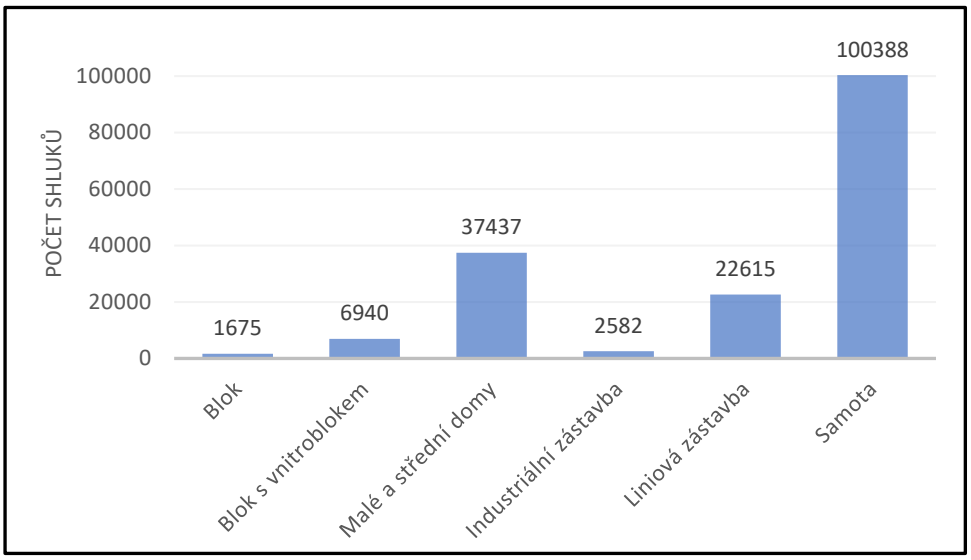
4.7.2.2 Výběr vstupních dat pro neuronovou síť

Proces přípravy vstupních dat lze v případě neuronových sítí rozdělit do dvou částí. Cílem prvního kroku je zjistit, zda je možné využít klasifikované shluky budov zástavby pomocí logistické regrese pro zvětšení trénovací množiny dat. Předpokladem je, že zvětšení trénovací množiny bude mít pozitivní dopad na výslednou přesnost klasifikace (Sun et al. 2017), neboť malá trénovací množina vede většinou k podprůměrné schopnosti modelu generalizovat, neboli je náchylnější k přeučení (Perez, Wang 2017). Druhým krokem je obdobně jako u logistické regrese proces výběru nejvhodnějších atributů.

Pro možnost využití dat z logistické regrese bylo nejprve nutné klasifikovat všech cca 170 000 shluků budov vzniklých pro segmentaci, včetně ručně vybrané trénovací množiny. Pro tento účel byl klasifikátor logistické regrese naučen na celé ručně vybrané trénovací množině. Klasifikátor byl nastaven pomocí parametrů zmíněných v kapitole 4.7.1.2. Pro klasifikaci bylo využito 13 charakteristik shluků vybraných v kapitole 4.7.1.1. Pomocí klasifikátoru byl všem 170 000 shluků budov přiřazen nejpravděpodobnější typ zástavby. Kromě nejpravděpodobnějšího typu bylo využito možnosti získat i hodnoty pravděpodobnosti zařazení daného shluku budov do jednotlivých tříd. Cílem zjištění této

informace je rozšířit trénovací množinu pouze o ty shluky budov, které jsou klasifikovány s určitou pravděpodobností.

Pro výběr shluků budov pro rozšíření trénovací množiny byla vytvořena funkce *SelectKBest (data, prob, k)*, která má na vstupu tři parametry. Prvním parametrem jsou vstupní data, v tomto případě jednotlivé shluky budov, včetně charakteristik a hodnot pravděpodobnosti zařazení k jednotlivým typům zástavby. Druhý parametr udává hodnotu pravděpodobnosti, která slouží pro vybrání nejvhodnějších dat. Poslední parametr určuje, jak se mohou lišit počty vybraných shluků budov náležící jednotlivým třídám.



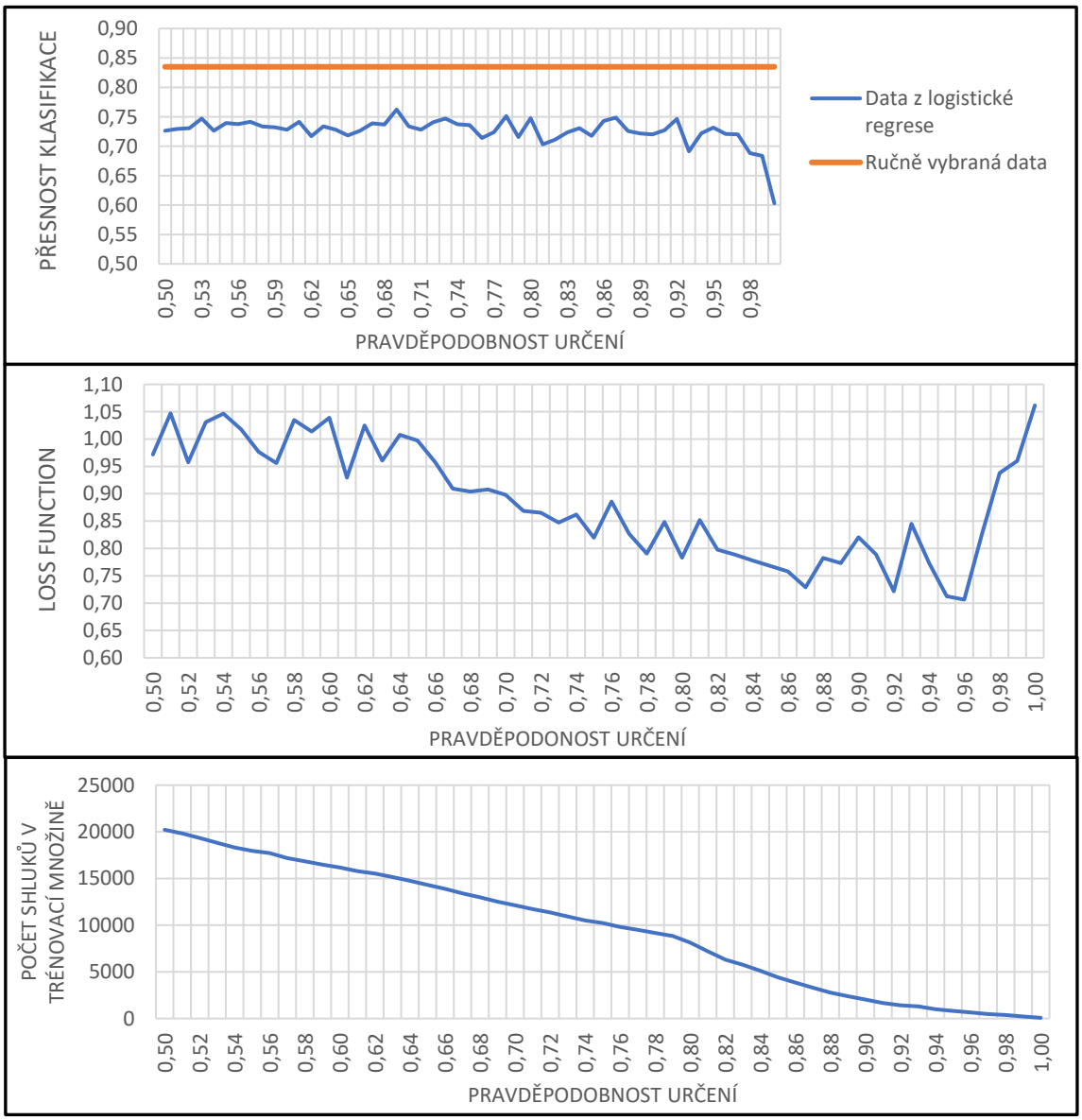
Graf 4: Rozložení počtů shluků budov po klasifikaci pomocí LR
Zdroj: *vlastní zpracování*

Před využitím zmíněné funkce byly ze vstupních dat odstraněny ty shluky budov, jejichž typ byl určen ručně. Tyto ručně klasifikované shluky budov budou sloužit jako testovací množina pro ověřování přesnosti klasifikace. Zbylé shluky budov následně vstupují do funkce *SelectKBest*. Nejdříve jsou pro každý typ zástavby vybrány ty shluky budov, které splňují podmínky pravděpodobnosti určení. Daný shluk budov splňuje tuto podmínku v případě, že hodnota atributu udávající pravděpodobnost zařazení daného shluku budov k určitému typu zástavby je vyšší jak hodnota *prob* vstupující do funkce *SelectKbest*. Jak lze vidět na grafu č. 4, rozložení počtu shluků budov v jednotlivých typech zástavby je po klasifikaci pomocí logistické regrese značně nevyrovnané. Proto lze očekávat, že nevyrovnané budou i počty vybraných shluků budov splňující podmínky pravděpodobnosti určení. Pro proces učení neuronové sítě je však vhodné, aby počty zástupců jednotlivých tříd byly přibližně vyrovnané (Mazurowski et al. 2008). Z tohoto důvodu je využito již zmíněné proměnné *k*. Nejprve jsou zjištěny počty vybraných

(splňujících podmínku pravděpodobnosti určení) shluků budov náležícím jednotlivým typům zástavby. Maximální počet vybraných shluků budov pro jednotlivé typy je poté roven k násobku počtu shluků budov v nejméně zastoupeném typu zástavby mezi vybranými shluky budov. Matematicky lze tento vztah vyjádřit jako:

$$N_i^{max} = k * \min(N_0, N_1, \dots, N_{n-1}, N_n)$$

kde N_i je počet vybraných shluků budov v jednotlivých typech zástavby a N_i^{max} je maximální počet shluků budov, které mohou být obsažené ve výběru pro daný typ. Pokud je $N_i^{max} < N_i$, je z vybraných shluků pro daný typ vybráno N_i^{max} shluků budov s nejvyšší hodnotou pravděpodobnosti určení. Pokud je $N_i^{max} > N_i$, je použito všech N_i vybraných shluků budov.



Graf 5: Využití dat z klasifikátoru LR pro trénování neuronové sítě
Zdroj: *vlastní zpracování*

Při stanovení proměnné *k*=1, by byly všechny třídy v trénovací množině stejně zastoupeny. Výhodou vyšší hodnoty *k* je, že pro trénovaní neuronové sítě je využito více prvků (charakteristik shluků budov). Nevýhodou je ovšem vzrůstající nevyrovnanost počtu shluků budov náležící jednotlivým typům zástavby. Z tohoto důvodu byla zvolena pro účely této diplomové práce hodnota *k*=3.

Vzhledem k tomu, že žádná data nejsou obsažena zároveň v trénovací a testovací části, není nutné pro ověření přesnosti klasifikace provádět *cross-validation* test. Kvůli prvku nahodilosti je však vhodné, provést vícenásobné natrénování sítě na stejných vstupních datech. Pro zjištění ideální hodnoty pravděpodobnosti, byla funkce *SelelectBest* opakovaně spuštěna v cyklu, kdy byla postupně měněna hodnota pravděpodobnosti určení (*prob)* od hodnoty 0,5 až po 1,0 s krokem 0,02. Neuronová síť byla vždy pro stejnou hodnotu pravděpodobnosti určení naučena celkem 5x. Před každý učením jsou ručně klasifikované shluky náhodně rozděleny na dvě poloviny. Vypočtené charakteristiky jedné poloviny slouží jako validační data pro kontrolu přesnosti klasifikace během procesu učení. Spočtené charakteristiky druhé poloviny ručně klasifikovaných shluků budov poté slouží pro otestování neuronové sítě. Při testování neuronové sítě je vypočítána přesnost klasifikace jako poměr správně klasifikovaných shluků budov z testovací množiny ku všem shlukům budov v testovací množině. Kromě přesnosti klasifikace je pomocí testovací množiny dat vypočtena též hodnota *Loss Function.* Tyto hodnoty byly zprůměrovány pro každou hodnotu pravděpodobnosti určení. Pro každou hodnotu pravděpodobnosti určení byl též zaznamenám počet shluků budov použitých pro trénování. Výsledné hodnoty lze vidět v grafu č. 5.

Pro porovnání přesnosti klasifikace při rozšíření trénovací množiny oproti přesnosti klasifikace při využití pouze ručně klasifikovaných shluků, je v grafu č. 5 zanesena hodnota přesnosti klasifikace nejvhodnější kombinace hyperparametrů sítě, získaná pomocí *cross-validation* testu při ladění hyperparametrů neuronové sítě.

Z grafu č. 5 lze vyčíst, že jakékoliv zvětšení trénovací množiny nepřineslo žádný pozitivní efekt pro zvýšení přesnosti klasifikace. Nejvyšší přesnosti bylo dosaženo při zvětšení trénovací množiny o shluky budov s pravděpodobností určení 0,7, kdy přesnost klasifikace dosáhla přibližně 76 %. Při této hodnotě pravděpodobnosti určení čítá trénovací množina přibližně 12 500 shluků budov. Pro ostatní hodnoty pravděpodobnosti určení shluku budov pomocí logistické regrese osciluje přesnost klasifikace pomocí neuronové sítě mezi 70-75 %. K výraznějšímu poklesu přesnosti

klasifikace dochází až při výběru shluků budov s pravděpodobností určení nad 0,93. Tento výrazný pokles je způsobem již relativně malým počtem shluků budov pro trénování neuronové sítě. Trochu odlišný pohled na využití dat z logistické regrese poskytuje křivka vývoje hodnoty *Loss Function* při různých hodnotách pravděpodobnosti určení. Jak lze z grafu vyčíst, hodnota *Loss Function* s rostoucí pravděpodobností určení shluku budov klasifikátorem logistické regrese postupně klesá. To znamená, že s klesajícím počtem shluků budov v trénovací množině dochází k tomu, že neuronová síť klasifikuje jednotlivé shluky budov s vyšší jistotou, ale bez vyšší hodnoty přesnosti klasifikace. Nárůst hodnoty *Loss Function* lze pozorovat až v případě, kdy trénovací množina čítá přibližně 800 shluků budov.

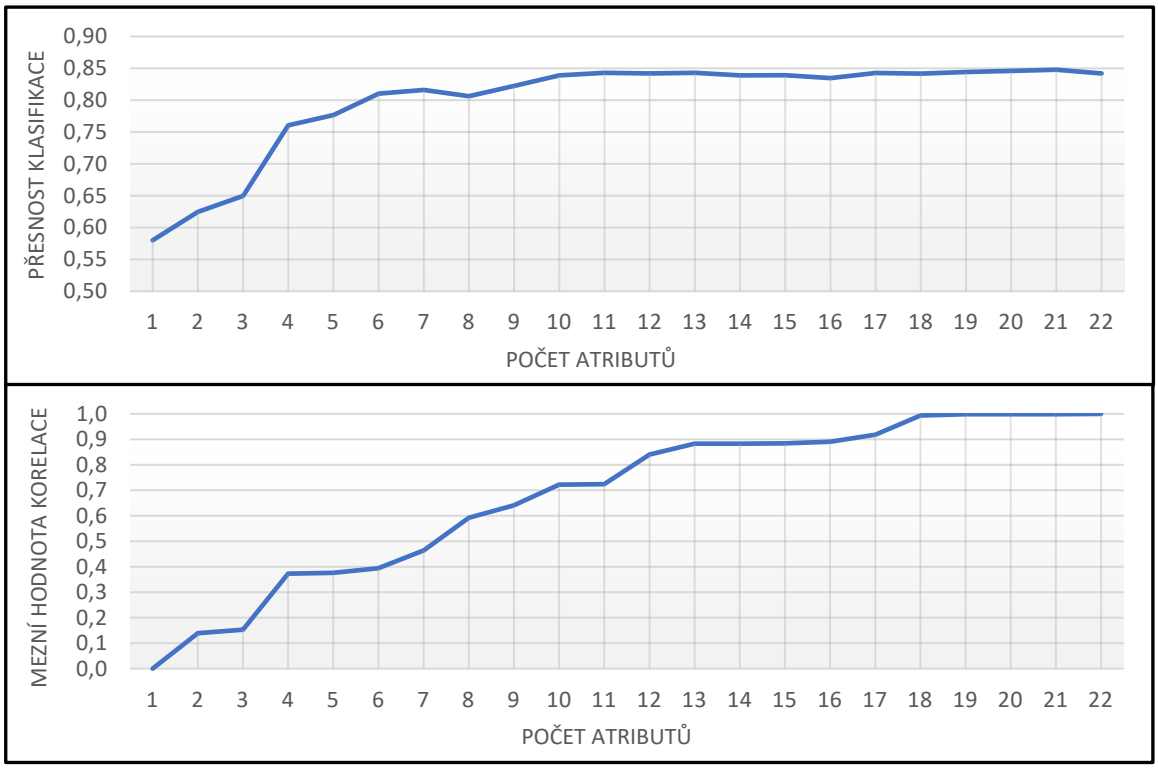
Druhou částí přípravy vstupních dat pro neuronové sítě je výběr nejvhodnějších atributů. Obdobně jako v případě výběru dat pro klasifikátor logistické regrese byly implementovány tři metody. První metoda je založena na výpočtu korelace jednotlivých charakteristik a postupném odstraňování nejvíce korelujících atributů. Druhá metoda používá pro výběr atributů ANOVA test. Třetí metoda využívá PCA analýzu. Podrobněji je fungování metod popsáno v kapitole č. 2.6.

1. **Výběr atributů na základě korelace**

   Obdobně jako v případě implementace této metody pro potřeby logistické regrese v kapitole 4.7.1.2 byl pro nalezení mezní hodnoty korelace, při které jsou vybrány nejvhodnější atributy, využit cyklus. Uvnitř tohoto cyklu dochází k postupné změně této mezní hodnoty. Pro zjištění úspěšnosti klasifikace pomocí atributů vybraných při určité míře korelace byl opět použit *cross-validation* test. Vstupními daty do tohoto testu je ručně vybraná trénovací množina shluků. *Cross-validation* test byl implementován stejně, jako při hledání optimálních hyperparametrů neuronové sítě v první částí této podkapitoly. Výsledky této metody lze vidět na grafu č. 6.

   V případě výběru atributů pro neuronovou síť bylo nejvyšší klasifikační přesnosti dosaženo při zachování 17 atributů. Tento počet atributů odpovídá mezní hodnotě korelace 0,92. Přesnost klasifikace při využití tohoto počtu dosahuje hodnoty přibližně 85 %. Avšak již od ponechání 11 a více atributů dosahuje přesnost klasifikace stále přibližně 84 %. Vzhledem k tomu, že doba provedení *cross-validation* testu není v případě neuronových sítí tak závislá na počtu atributů, bylo by nejvhodnější při

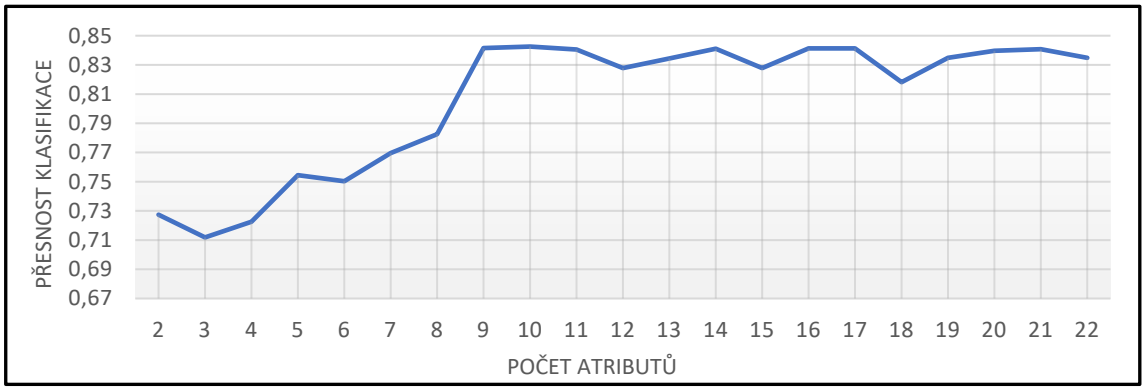
využití této metody nastavit mezní hodnotu korelace na hodnotu 0,918, neboli využít 17 popisných charakteristik.



Graf 6: Výběr atributů pro NN na základě korelace
Zdroj: *vlastní zpracování*

2. Výběr atributů pomocí ANOVA testu

Druhá metoda pro výběr atributů byla implementována opět s využitím knihovny *Scikit-learn*. Na rozdíl od předchozí metody, vstupuje do této metody požadovaný počet vybraných charakteristik. Tato metoda byla opět implementována pomocí cyklu, kdy byl postupně zvyšován počet požadovaných atributů od jednoho až po maximálních 22 charakteristik. Stejně jako v první metodě byl pro ověření úspěšnosti klasifikace využit *cross-validation* test, tak jak byl implementován pro hledání hyperparametrů neuronové sítě. Výsledky této metody lze vidět na grafu č. 7.

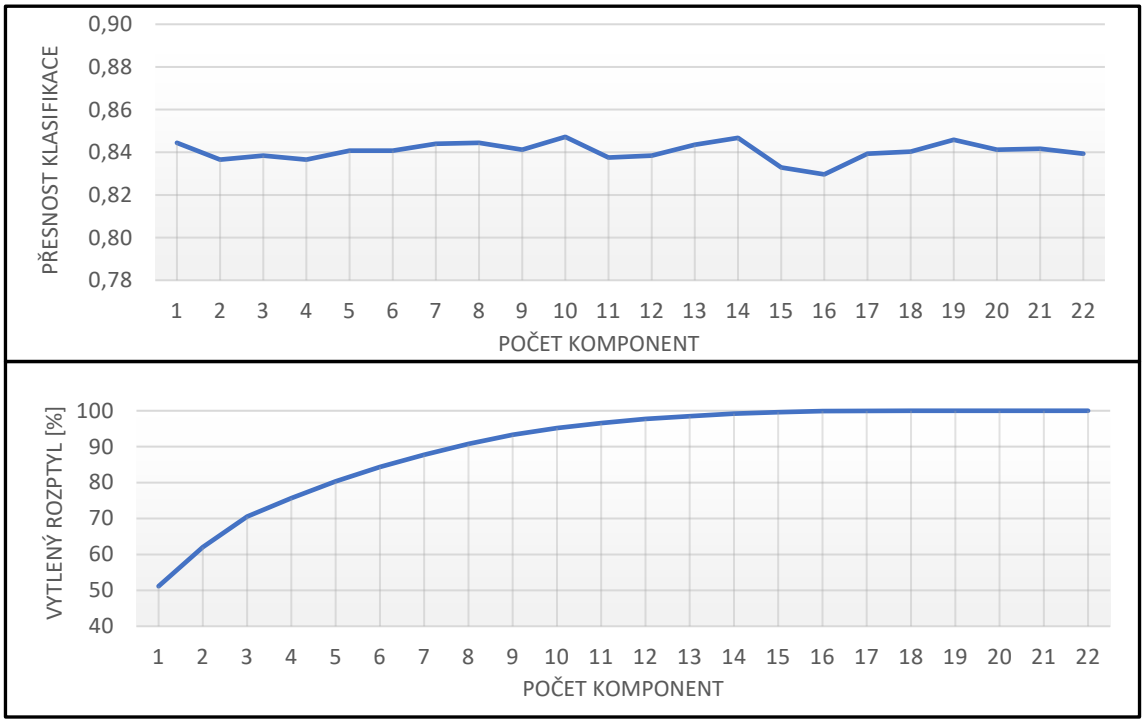


Graf 7: Výběr atributů pro NN pomocí ANOVA test
Zdroj: *vlastní zpracování*

Pro tuto metodu byla maximální hodnota přesnosti klasifikace zaznamenána již při využití 14 atributů z celkových 22. Dosažená hodnota přesnosti klasifikace je v tomto případě 84 %. Průměrné hodnoty přesnosti klasifikace okolo 83 % je dosaženo téměř vždy při využití více než 9 charakteristik.

3. **Výběr atributů pomocí metody založená na PCA**

I pro implementaci této metody byla využita knihovna *Scikit-learn*. Vstupem do funkce založené na metodě PCA je výsledný počet komponent, do kterých jsou vstupní charakteristiky transformovány. Pro nalezení počtu komponent, při kterých je dosaženo nejvyšší přesnosti klasifikace je tato funkce volána opět v cyklu. Počet komponent je zvyšován od 1 až po 22. Obdobně jako u předchozích dvou metod byl pro zjištění přesnosti klasifikace využit *cross-validation* test, tak jak byl implementován pro hledání hyperparametrů neuronové sítě. Výsledky této metody lze vidět na grafu č. 8. Na rozdíl od „výběrů atributů“ touto metodou pro logistickou regresi, byly v tomto případě naměřeny mnohem vyšší hodnoty přesnosti klasifikace. Z grafu lze vypozorovat, že hodnota přesnosti klasifikace je přibližně stejná pro jakýkoliv počet komponent, i přes skutečnost, že pomocí pouze jedné komponenty je vysvětleno jen 50 % veškeré variability. Maximální hodnota přesnosti klasifikace byla dosažena při transformaci vstupních 22 charakteristik do 10 komponent. Při tomto počtu komponent bylo dosaženo přesnosti klasifikace 84,5 %.



Graf 8: Výběr atributů pro NN na základě korelace
Zdroj: *vlastní zpracování*

Všechny tři metody dosáhly velice podobných výsledků. Přesnost klasifikace byla, oproti využití všech 22 vstupních charakteristik, pomocí metod zvýšena o 1-2 %. Jak již bylo řečeno dříve, čas učení neuronové sítě není téměř závislý na množství vstupních atributů. Z tohoto důvody nepřináší metody výhodu snížení časové náročnosti při využití nižšího počtu charakteristik. Z důvodu dosažení nepatrně vyšší přesnosti klasifikace byla pro další využití v práci zvolena metoda založená na korelaci. Vybraných 17 charakteristik lze vidět v tabulce č. 10.

Vybrané charakteristiky		
Součet plochy budov	Průměrná plocha budovy	Součet obvodů budov
Počet budov	Obvod konvexní obálky	Poměr plochy budov k ploše polygonu z první části segmentace
Poměr šířky a délky opsaného obdélníku s minimální šířkou	Směrodatná odchylka v ploše budov	Směrodatná odchylka ve směru
Průměrná vzdálenost k nejbližší budově	Průměrná vzdálenost ke třem nejbližším budovám	Průměrná výška budovy
Průměrná vzdálenost k silnici	Průměrný obvod budovy	Poměr plochy budov ku ploše konvexní obálky
Plocha konvexní obálky	Šířka opsaného obdélníku s minimální šířkou	

Nevybrané charakteristiky		
Šířka opsaného obdélníku s minimální šířkou	Délka opsaného obdélníku s minimální plochou	Plocha opsaného obdélníku s minimální plochou
Délka opsaného obdélníku s minimální šířkou	Plocha opsaného obdélníku s minimální šířkou	

Tabulka 10: Seznam výsledných atributů pro NN
***Zdroj**: vlastní zpracování*

4.7.2.3 Analýza výsledků klasifikace pomocí neuronové sítě

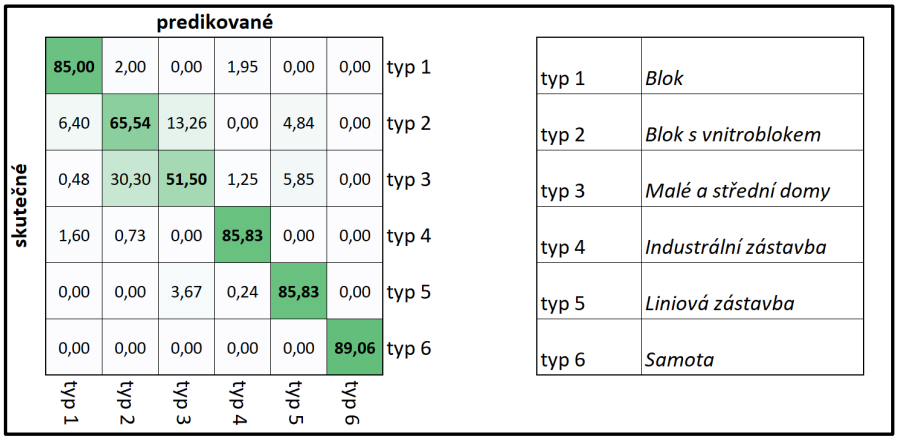
Cílem této kapitoly práce je poskytnout bližší pohled na výsledky klasifikace pomocí neuronové sítě využívající charakteristiky shluků budov. Pro posouzení výkonnosti klasifikátoru byla opět zvolena chybová matice, respektive metriky z ní vycházející (kapitola 2.7).

Pro sestrojení výsledné chybové matice byl využit velice podobný postup jako při hledání ideálních charakteristik v kapitole 4.7.2.2. Vzhledem k tomu, že rozšíření

trénovací množiny pomocí dat z logistické regrese nepřineslo kýžené výsledky, byla jako trénovací množina pro finální zhodnocení přesnosti klasifikace zástavby pomocí neuronové sítě zvolena ručně vybraná trénovací množina. Z celkových 22 charakteristik bylo pro závěrečnou analýzu výkonnosti klasifikátoru využito 17 vybraných pomocí metody založené na korelaci (tabulka 10). Pro nastavení neuronové sítě byly využity optimalizované hyperparametry z kapitoly 4.7.2.1 (tabulka 9).

Pro sestrojení chybové matice byl využit *cross-validation* test. Aby mohlo být využito více dat pro otestování a validaci, byla data z ruční trénovací množiny rozdělena na 4 stejně početné části (*k=4)*. Více prvků ve validační části zajistí, že počítaná hodnota *Validation Loss* bude mít vyšší vypovídající hodnotu. Díky tomu funkce *EarlyStopping* může zastavit proces učení ve vhodnější chvíli. Vyšší počet prvků v testovací části dat poté zajistí, že přesnost klasifikace spočtená pomocí testovací části bude více relevantní. Je nutné však nezmenšit trénovací část příliš, neboť méně dat v trénovací části může způsobit, že model (natrénovaná neuronová síť) může poskytovat velice rozdílné hodnoty přesnosti klasifikace v závislosti na použitých testovacích datech (Beleites et al. 2005). Při rozdělování na jednotlivé části je zajištěno, že rozložení počtu shluků budov náležící jednotlivým kategoriím v jednotlivých částech je shodné, jako rozložení počtu shluků budov v celé ručně vybrané trénovací množině. Aby byl alespoň částečně odstraněn prvek náhody při trénování neuronové sítě, je v každé iteraci *cross-validation* testu neuronová síť naučena celkem 8x. To znamená, že pro závěrečné určení výkonnosti klasifikace pomocí neuronové sítě, je neuronová síť naučena celkem 32x. Před každým učením neuronové sítě je část určená pro testování náhodně rozdělena na dvě shodně velké části. Jedna z těchto částí je poté použita pro validaci při trénování sítě a jedna pro sestavení chybové matice (obrázek 39). Všechny chybové matice byly sečteny a následně vyděleny ½ počtu učení neuronové sítě při jedné iteraci *cross-validation* testu. Z toho vyplývá, že součet hodnot v chybové matici je roven počtu shluků budov ručně vybrané trénovací množiny. Čas nutný pro naučení a otestování jedné neuronové sítě v prostředí *Google Colab* byl v tomto případě přibližně 30 sekund. Výslednou chybovou matici lze vidět na obrázku č. 37. Metriky pro hodnocení výkonnosti klasifikátoru jsou poté zaneseny v tabulce č. 11.

Jak lze vyčíst z tabulky č. 11, výsledná přesnost klasifikace zástavby pomocí neuronových sítí dosáhla hodnoty 86,5 %. To znamená, že pomocí metod na výběr nejvhodnějších atributů a zvýšení počtu validačních dat, se podařilo zvýšit přesnost klasifikace o 3 % oproti případu, kdy bylo pro klasifikaci využito všech 22 charakteristik.



Obrázek 37: Chybová matice pro klasifikaci zástavby pomocí NN
***Zdroj**: vlastní zpracování*

Nejlépe klasifikovaným (*F-score = 1)* typem zástavby při klasifikaci pomocí neuronových sítí byl typ *Samota*. Při všech 32 testování neuronové sítě byly všechny shluky budov tohoto typu zástavby klasifikovány správně a žádný ze shluků náležících zbylým typům zástavby nebyl označen jako typ *Samota*. Velice dobře (*F-score = 0.968)* byl klasifikován též typ *Industriální zástavba*. Hodnota *F-score* vyšší než 0.9 byla naměřena též u typů *Blok* a *Liniová zástavba*. Oba typy zástavby byly klasifikátorem lehce nadhodnocovány (*Recall* > *Precision*).

Typ zástavby	Precision	Recall	F-score
Blok	0,909	0,956	0,932
Blok s vnitroblokem	0,665	0,728	0,695
Malé a střední domy	0,753	0,576	0,653
Industriální zástavba	0,961	0,974	0,967
Liniová zástavba	0,889	0,956	0,922
Samota	1,000	1,000	1,000
		Accuracy	**0,864**

Tabulka 11: Analýza přesnosti klasifikace zástavby pomocí NN
***Zdroj**: vlastní zpracování*

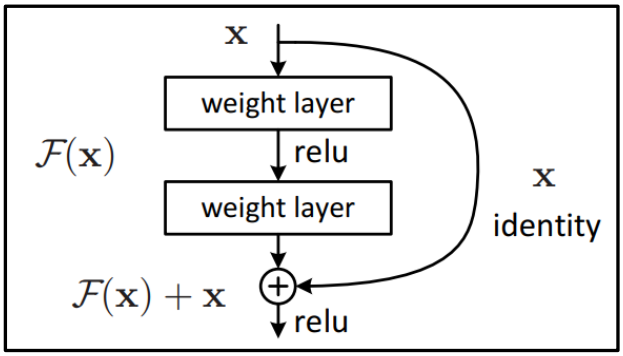
O poznání hůře byly klasifikovány zbylé dva typy zástavby *Blok s vnitroblokem* a *Malé a střední domy*. Hodnota *F-score* byla pro oba typy již nižší než 0.7. Zejména typ *Malé a střední domy* byl ve třetině případů zaměněn klasifikátor za typ *Blok s vnitroblokem.*

4.7.3 Konvoluční neuronová síť

Cílem této části práce je představit implementaci klasifikátoru založeného na konvolučních neuronových sítí pro klasifikaci zástavby pomocí rastrových podob shluků budov. V první části této kapitoly je popsán proces hledání nejvhodnější architektury sítě a ladění hyperparametrů sítě. Následně je diskutováno využití dat z klasifikátoru logistické regrese pro zvětšení trénovací množiny. Poslední část je věnována analýze výsledků tohoto klasifikátoru s ohledem na přesnost klasifikace jednotlivých typů zástavby.

4.7.3.1 Architektura konvoluční neuronové sítě a hledání hyperparametrů

Jak již bylo zmíněno v kapitole 4.7.2.1, pro zmenšení prohledávaného prostoru hyperparametrů je vhodné, vytvořit takovou architekturu sítě, která bude využívat více opakujících se bloků vrstev. Pro účely této diplomové práce byly vybrány dvě různé architektury CNN.

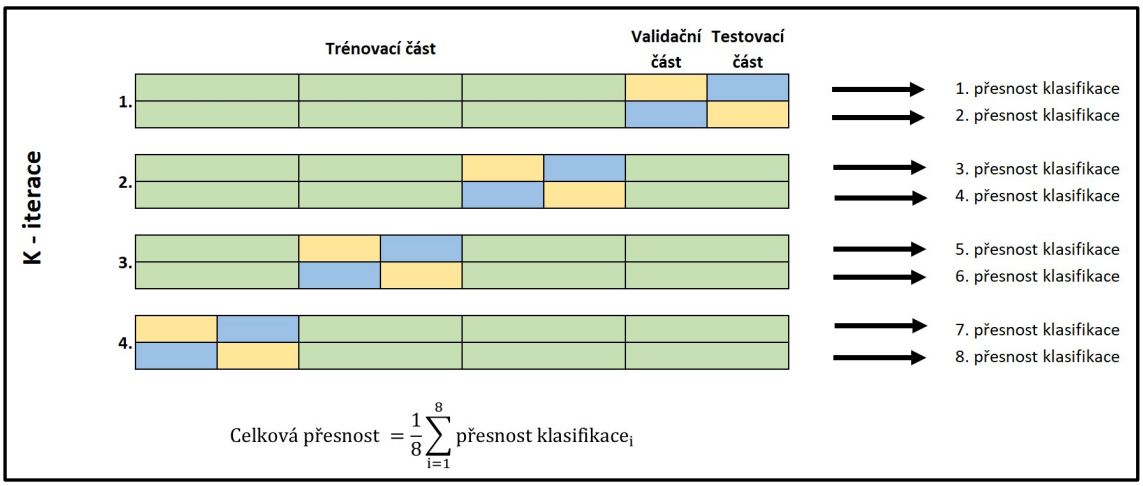


Obrázek 38: Reziduální blok CNN ResNet
***Zdroj**: He et al. (2016)*

1. **ResNet**

První architektura je zástupcem složitějšího typu architektury neuronových sítí (obrázek 16, vpravo), kdy jednotlivé vrstvy mohou mít více než jeden vstup. Konkrétně se jedná o neuronovou síť ResNet (He et al. 2016). Tato CNN patří mezi nejpoužívanější architektury v oblasti počítačového vidění a rozpoznávání obrazu. Základním stavebním kamenem této sítě jsou takzvané reziduální bloky (obrázek 38). Oproti běžnému řetězení vrstev je v tomto bloku využito takzvaných zkratkových spojeních (*skip connections*). V případě reziduálních bloků je na výstupu kombinován vstup do bloku (*x*) a vstup do bloku, který prošel souborem několika vážících vrstev (*F(x)*). Vážící vrstvy jsou v tomto případě představovány konvolučními neuronovými vrstvami.

Pro prvotní otestování vhodnosti této architektury byla architektura převzata z originálního článku (He et al. 2016), včetně většiny hodnot hyperparametrů jako jsou počty a velikost konvolučních filtrů v konvolučních vrstvách. Parametry, které z originálního návrhu nebyly převzaty jsou: počet reziduálních bloků, *Learning rate* a *Batch size*. Důvodem pro nepřevzetí hodnot těchto parametrů je fakt, že hodnoty těchto parametrů jsou velmi ovlivněny velikostí trénovací množiny. Optimalizátor byl též převzat z originálního článku. Daným optimalizátorem je obdobně jako u neuronové sítě *Adam*. Schéma architektury CNN *ResNet* při využití jediného reziduálního bloku lze vidět na obrázku č. 40. Bližší popis jednotlivých vrstev této neuronové sítě lze nalézt v kapitole č. 2.4.



Obrázek 39: Cross-validation test pro CNN
***Zdroj**: vlastní zpracování*

Pro implementaci této CNN byla opět využita knihovna *Tensorflow 2.0* a vývojové prostředí *Google Colab*. Pro určení vhodnosti architektury této CNN byl opět použit *cross-validation* test. Pro tento účel byla vstupní data a ručně vybraná trénovací množina představovaná rastrovými podobami shluků, náhodně rozdělena na 4 stejně početné částí ($K = 4$). I v tomto případě byla aplikována podmínka stejného rozložení počtu shluků náležícím jednotlivým typům v jednotlivých částech a v celé ručně vybrané trénovací množině. Obdobně jako u neuronové sítě v kapitole 4.7.2 byla každá část určená pro testování rozdělena na dvě stejné velké části. Jedna část poté slouží pro validaci během učení. Druhá část slouží pro otestování sítě a získání hodnoty přesnosti klasifikace pomocí určení poměru správně klasifikovaných shluků ku všem klasifikovaným shlukům. Schéma *cross-validation* testu použitého pro tento účel lze vidět na obrázku č. 39.

Vzhledem k mnohonásobně delšímu času učení CNN oproti běžné neuronové síti bylo nutné snížit celkový počet učení CNN během *cross-validation* testu. V tomto případě, je neuronová síť při každé iteraci *cross-validation* testu naučena pouze 2x. Při druhém učením sítě je zaměněna testovací část za validační. Výsledky přesnosti klasifikace jsou poté zprůměrovány přes všechna testování, jak to lze vidět na obrázku č. 39. Celý *cross-validation* test je poté zopakován. Před druhým testem jsou data opět náhodně rozdělena na 4 části. Díky tomu oba testy proběhnou na odlišně rozdělených vstupních datech. Pro částečné urychlení procesu učení byla obdobně jako u neuronových sítí implementována funkce *EarlyStopping*. Maximální počet epoch byl pro tento účel stanoven na 200.

Oproti práci s klasickou neuronovou sítí, kdy jsou data vstupující do procesu trénování a testování uchovávána nejčastěji ve dvourozměrných polích, je pro CNN vhodné mít vstupní rastry rozděleny pomocí stromové struktury složek, ze kterých jsou poté jednotlivé rastry načítány. Před každým učením CNN jsou vstupní data rozdělena do tří složek: *Train, Validation, Test*. Uvnitř z každé těchto složek jsou data dále rozdělena opět do několika dalších složek, které odpovídají klasifikovaným typům zástavby.

Vrstva	Parametry	Popis
Konvoluční	32, 3x3	Úvodní vrstvy
Aktivační funkce	ReLu	
Konvoluční	32, 3x3	
Aktivační funkce	ReLu	
Max Pooling	3x3	
Konvoluční	64, 3x3	Reziduální blok
Aktivační funkce	ReLu	
Batch Normalization		
Konvoluční	64, 3x3	
Batch Normalization		
Aktivační funkce	ReLu	
Konvoluční	64, 3x3	Závěrečné vrstvy
Aktivační funkce	ReLu	
Global Average Pooling		
Dense	256	
Dropout	0,5	
Dense	6	

Obrázek 40: Architektura sítě ResNet
***Zdroj**: He et al. (2016), upraveno*

Vzhledem k tomu, že prohledávaný prostor zmíněných tří hyperparametrů neuronové sítě *ResNet* není příliš velký, byla pro nalezení nejvhodnější kombinace

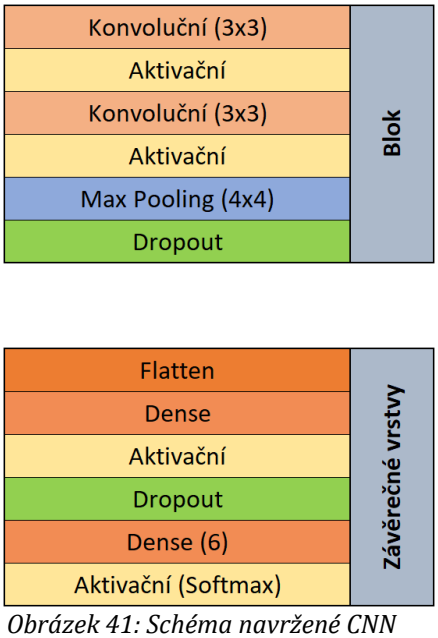
využita zjednodušená podoba metody *GridSearch*, která byla implementována pomocí tří vnořených *for* cyklů. Hodnoty, kterých hledané hyperparametry mohou nabývat jsou zaneseny v tabulce č. 12. Ve stejné tabulce je též uvedena nejvhodnější kombinace daných hodnot. Výsledná přesnost klasifikace při využití nejvhodnější kombinace hodnot hyperparametrů dosáhla hodnoty přibližně 70 %.

Hyperparametr	Testované hodnoty	Nejvhodnější hodnota
Počet reziduálních bloků	[1, 3, 5, 10, 15, 20, 30]	1
Learning rate	[0,0001;0,001;0,01;0,1]	0,0001
Batch size	[16, 32, 64, 128]	32

Tabulka 12: Optimalizované hyperparametry sítě ResNet
***Zdroj**: vlastní zpracování*

2. **Řetězení vrstev**

Druhá architektura CNN využívá jednoduššího principu, kdy vstup do vrstvy je představován pouze výstupem vrstvy předcházející (obrázek 16, vlevo). Na základě literatury a řady experimentů byl vytvořen blok šesti vrstev. Jednotlivé vrstvy bloku lze vidět na obrázku č. 41, nahoře. Pro snížení množství volitelných hyperparametrů neuronové sítě mají konvoluční vrstvy pevně stanovenou velikost konvolučního filtru. Obdobně u *Max Pooling layer* je pevně stanovena velikost poolingového okna. Navržená architektura CNN je uzavřena pomocí dalších 6 vrstev. Závěrečné vrstvy lze vidět na obrázku č. 41, dole.



Obrázek 41: Schéma navržené CNN
***Zdroj**: vlastní zpracování*

Pro sestavení funkční architektury neuronové sítě bylo nutné stanovit hodnoty několika dalších hyperparametrů. Pro konvoluční vrstvy je nutné určit počet filtrů. Tato hodnota může být unikátní pro každou konvoluční vrstvu v síti. Dále je též nutné určit počet neuronů v první *Dense layer* ve skupině závěrečných vrstev. Dalším hyperparametrem, který je nutné stanovit, je volba aktivační funkce. Obdobně jako u neuronové sítě v kapitole 4.7.2 je aktivační funkce volena jednotně pro celou síť. Předposledním hyperparametrem souvisejícím s architekturou sítě je výše *dropoutu.* Tato hodnota může být opět unikátní pro každou *Dropout layer*. Posledním volitelným hyperparametrem týkající se vrstev CNN je využití *L2 regularizace*. Princip regularizace byl popsán v kapitole 2.4. Hodnota regularizace (0,0001) byla převzata z další velmi používané architektury CNN pro rozpoznávání obrazu – *Inception V3* (Szegedy et al. 2015). Pro účely této práce jsou testovány dvě možnosti využití regularizace. První možností je použití regularizace u všech možných vrstev (*konvoluční* a *Dense*). Druhou možností je sestavení sítě bez použití regularizace. Zbývající hodnoty hyperparametrů, které je nutné stanovit jsou: velikost *Batch size*, hodnota *Learning rate* a volba *optimalizátoru*. Úplně posledním hyperparametrem je stanovení počtu navržených bloků, které budou v sítí použity. Celkový počet hyperparametrů je roven $7 + k * 3$, kde k je počet využitých bloků. Výčet jednotlivých hyperparametrů, včetně hodnot, kterých mohou jednotlivé hyperparametry nabývat je zanesen v tabulce č. 13.

Hyperparametr	Testované hodnoty
Počet filtrů konvoluční vrstvy	[16,32,48,64,96,128,256,512]
Počet neuronů Dense vrstvy	[16,32,64,128,256,512,1024]
Velikost Dropoutu	(0;1)
Aktivační funkce	[ReLU,Sigmoid]
L2 regularizace	True/False
Learning rate	[0,0001; 0,001; 0,01; 0,1]
Optimalizátor	[SGD, RMSProp, Adam, Adagrad, Adamax]
Batch size	[16,32,64,128]
Počet bloků	<1;10>

Tabulka 13: Testovaný rozsah hyperparametrů pro navrženou CNN
***Zdroj**: vlastní zpracování*

Vzhledem k relativně velkému množství hyperparametrů, které je nutné určit, byla pro účely této práce zvolena *bayesovská optimalizační* metoda. Pro implementaci celé CNN byla opět využita knihovna *Tensorflow 2.0* a vývojové prostředí *Google Colab*. Pro implementaci optimalizační metody to poté byla knihovna *Optuna*. Pro posouzení vhodnosti dané kombinace vstupních parametrů byl použit stejný algoritmus založený na *cross-validation* testu, jako v případě hledání ideálních hyperparametrů pro síť *ResNet* dříve v této kapitole. Celkově bylo otestováno přibližně 1000 kombinací hodnot jednotlivých hyperparametrů. Celkový čas hledání ideální kombinace hodnot hyperparametrů byl více než 100 hodin. Nejlepší nalezenou kombinaci hodnot hyperparametrů a zároveň výslednou architekturu neuronové sítě, lze vidět na obrázku č. 42. Přesnost klasifikace nejlepší nalezené kombinace dosáhla přibližně 85 %. Vzhledem k tomu, že tato hodnota je mnohem vyšší než v případě CNN *ResNet*, byla tato navržené architektura neuronové sítě vybrána pro další využití v této diplomové práci.

Architektura
Konvoluční (64, 3x3), L2 regularizace
Aktivační (ReLu)
Konvoluční (128, 3x3), L2 regularizace
Aktivační (ReLu)
Max Pooling (4x4)
Dropout (0,3)
Konvoluční (128, 3x3), L2 regularizace
Aktivační (ReLu)
Konvoluční (256, 3x3), L2 regularizace
Aktivační (ReLu)
Max Pooling (4x4)
Dropout (0,3)
Flatten
Dense (512), L2 regularizace
Aktivační (ReLu)
Dropout (0,3)
Dense (6)
Aktivační (Softmax)

Ostatní hyperparametry
Learning rate = 0,0001
Batch size = 32
Optimalizátor = ADAM

Obrázek 42: Výsledná architektura navržené CNN
***Zdroj**: vlastní zpracování*

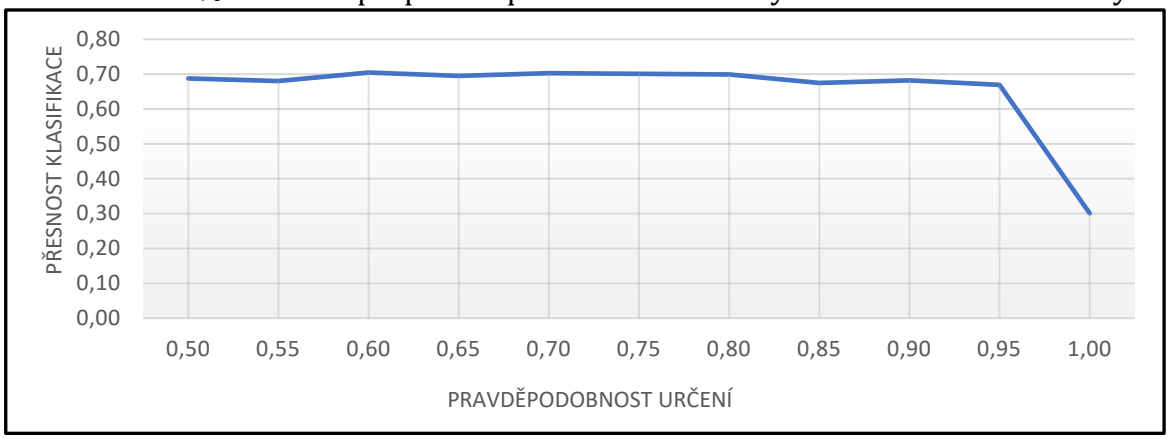
4.7.3.2 Úprava trénovací množiny

V této kapitole jsou diskutovány dva způsoby pro úpravu trénovací množiny. Prvním způsobem je zvětšení trénovací množiny pomocí klasifikovaných shluků budov klasifikátorem logistické regrese. Druhým způsobem je implementace procesu zvaného datová augmentace. Cílem obou způsobů je zvýšit počet shluků budov určených pro trénování CNN. Pro neuronové sítě určené pro rozpoznávání obrazu je doporučeno

použití minimálně 1000 prvků trénovací množiny na jednu třídu (Warden 2017). To je několikanásobně více, než je aktuální počet ručně vybraných trénovacích shluků budov. Obě metody mají i svou nevýhodu. A to konkrétně, že při nich nedochází k rozšíření trénovací množiny, v pravém slova smyslu, ale k nahrazení původní trénovací množiny pomocí jiných prvků. Pro účely ověření obou metod byla využita architektura sítě a její hyperparametry navržené v předchozí podkapitole (obrázek 42).

1. **Využití dat z logistické regrese**

Pro zjištění, zda je možné pomocí klasifikovaných shluků budov metodou klasifikátoru logistické regrese dosáhnout vyšší hodnoty přesnosti klasifikace, byl využit stejný algoritmus, jako v případě klasické neuronové sítě v kapitole 4.7.2.2. Pro výběr shluků s určitou pravděpodobností stanovené klasifikátorem logistické regrese byla využita již zmíněná funkce *SelectKBest*. Parametr *k* této funkce byl i v tomto případě roven 3. Daná funkce je volána v cyklu, kdy je postupně upravována hodnota pravděpodobnosti určení v intervalu (0,5;1) s krokem 0,05. Pro validaci a testování CNN byla využita ručně vybraná trénovací množina shluků budov. Pro danou hodnotu pravděpodobnosti určení byla neuronová síť naučena vždy celkem 4x. Pro každé učení neuronové sítě byla ručně vybraná trénovací množina náhodně rozdělena na dvě stejně velké části určená pro validaci a testování. Pomocích testovacích dat byla vždy spočtena přesnost klasifikace jako poměr správně klasifikovaných shluků budov ku všem klasifikovaným. Výsledná přesnost klasifikace pro danou hodnotu pravděpodobnosti určení je spočtena jako průměr všech 4 testovaní. Výsledky této metody je možné vidět v grafu č. 9. Jak lze z grafu vyčíst, využití klasifikovaných shluků klasifikátorem logistické regrese nemělo žádný pozitivní efekt. Výsledná přesnost klasifikace byla dokonce o 15 % nižší než při použití pouze dat z ručně vybrané trénovací množiny.



Graf 9: Využití dat z logistické regrese pro trénování CNN
Zdroj: *vlastní zpracování*

2. **Datová augmentace**

Datová augmentace je jeden ze způsobů, jak zvýšit schopnost neuronové sítě generalizovat, pomocí zvýšení počtu prvků v trénovací množině, pouze s využitím informací v ní již obsažených (Taylor, Nitschke 2017). Datová augmentace v případě CNN spočívá v geometrických, popřípadě barevných úprav vstupních rastrů. Vzhledem k tomu, že rastrové podoby shluků budov jsou pouze jednopásmovým binárním obrazem, přicházejí v úvahu pro tuto práci pouze operace geometrické.

Geometrická transformace	Rozsah
Rotace	0-360°
Přiblížení	-0,1 % – 0,1 %
Posunutí do stran	-0,1 % – 0,1 %
Posunutí nahoru/dolu	-0,1 % – 0,1 %
Vertikální převrácení	True/False
Horizontální převrácení	True/False

Tabulka 14: Data augmentation
***Zdroj**: vlastní zpracování*

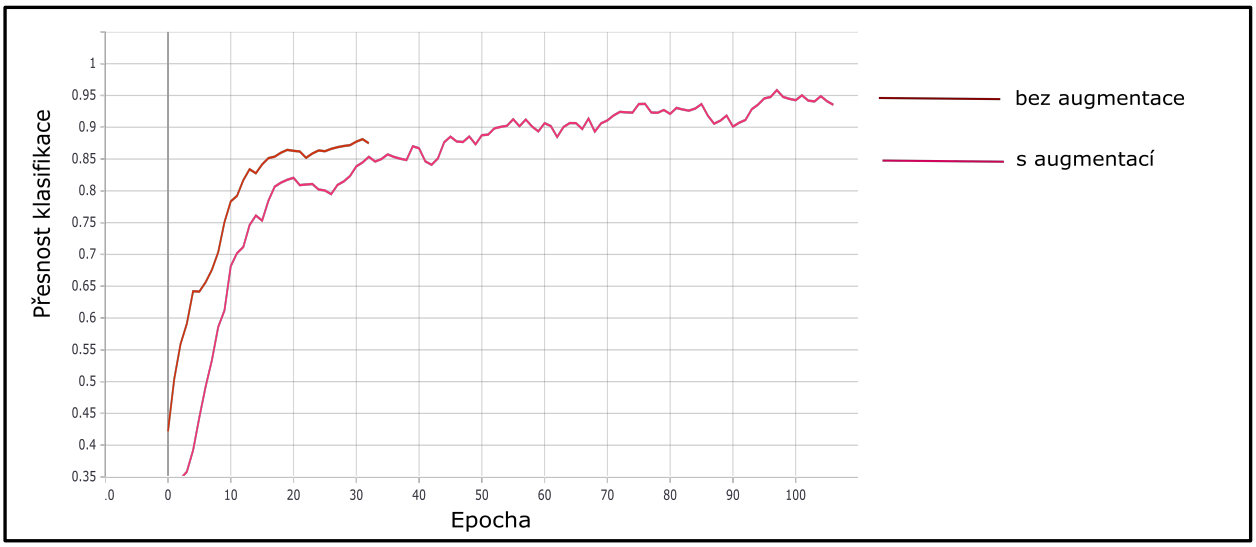
Pro implementaci datové augmentace byla využita funkce (*Image data generator*) z knihovny *Tensorflow 2.0*. Tato funkce dostane na vstupu skupinu rastrů. Na každý z těchto rastrů jsou přímo v paměti počítače aplikovány vybrané geometrické transformace. Upravené rastry poté rovnou vstupují do procesu učení. Každá geometrická operace je definována parametrem s určeným rozsahem. Při transformaci je vždy náhodně volena hodnota daného parametru z daného rozsahu. Cílem tohoto kroku je zaručit, že v průběhu učení vstupují do CNN takové rastry, které síť ještě neviděla. V případě, že datová augmentace není aplikovaná, je CNN v každé epoše trénována na stejných rastrech. V tabulce č. 14 je uveden seznam geometrických transformací včetně rozsahů hodnot parametru, které byly využity pro účely této práce. Pro určení přínosu datové augmentace pro navrženou CNN byl opět použit *cross-validation* test, tak jak byl implementován pro hledání hodnot hyperparametrů sítě v kapitole 4.7.3.1 (obrázek 39). Pro možnost co nejpřesnějšího porovnání vlivu datové augmentace na výslednou přesnost, je vždy neuronová síť učena 2x. Jednou na datech, která prošla datovou augmentací, a podruhé na originálních rastrech. Pro testování i validaci byla použita pro obě sítě úplně stejná data, tudíž bez použití datové augmentace. Pomocí testovacích dat byla vždy vypočítána přesnost klasifikace

a spočtena hodnota *Loss Function*. Kromě hodnot z testování sítě byl zaznamenán i počet epoch, nutných k naučení sítě. Naměřené hodnoty přesnosti klasifikace, *Loss Function* a počtu epoch byly na závěr zprůměrovány celkovým počtem učení CNN dané varianty. Výsledné hodnoty lze vidět v tabulce č. 15.

	Přesnost klasifikace	Loss Function	Počet epoch
Bez datové augmentace	85,3 %	0,78	35
S datovou augmentací.	89,1 %	0,79	60

Tabulka 15: Analýza využití datové augmentace
***Zdroj**: vlastní zpracování*

Z dané tabulky je patrné, že došlo k výraznému nárůstu přesnosti klasifikace rastrových podob shluků budov při využití datové augmentace. Hodnota *Loss Function* byla i přes vyšší dosaženou přesnost klasifikace přibližně stejná. Použití datové augmentace se též projevilo vyšším počtem epoch nutných k naučení sítě. Přesněji počtu epoch použitých k natrénování sítě do doby, než bylo trénování zastaveno funkcí *EarlyStopping.* Vyšší počet epoch je dán zejména pomalejším nárůstem přesnosti klasifikace v počátku průběhu učení, jak lze vidět na grafu č. 10.

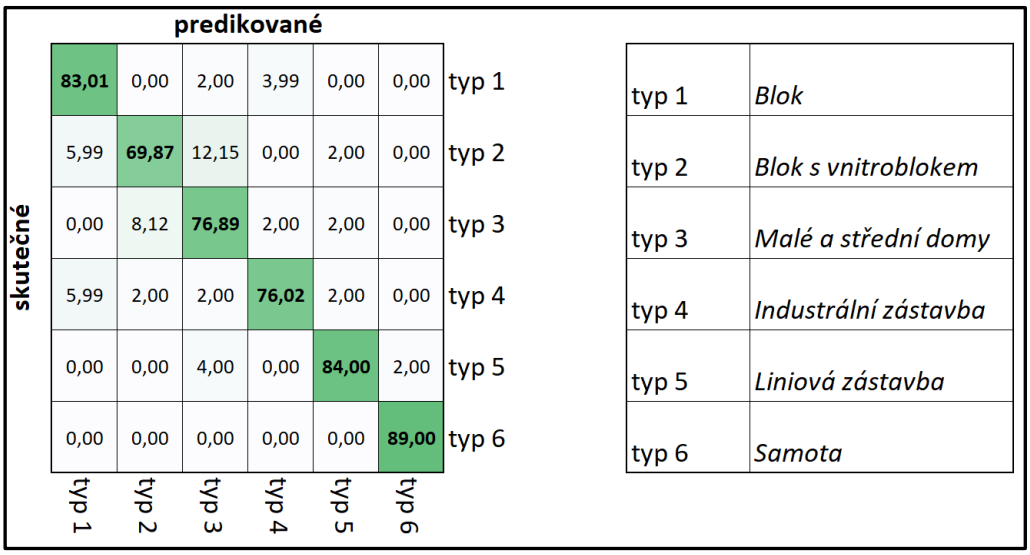


Graf 10: Využití datové augmentace
***Zdroj:** vlastní zpracování*

4.7.3.3 Analýza výsledků klasifikace zástavby pomocí CNN

Pro závěrečné zhodnocení výkonnosti klasifikátoru CNN byla využita architektura sítě navržená v kapitole 4.7.3.1, včetně hyperparametrů určených ve stejné kapitole. Ručně vybraná trénovací množina byla pro závěrečnou analýzu upravena pomocí metody datové augmentace.

Pro posouzení výkonnosti klasifikátoru byla opět zvolena chybová matice, respektive metriky z ní vycházející (viz kapitola 2.7). Hodnoty uvnitř chybové matice byly získány pomocí *cross-validation* testu, tak jak byl implementován pro hledání ideálních hyperparametrů neuronové sítě v kapitole 4.7.3.1. Testovací množina v tomto případě neslouží pro výpočet přesnosti klasifikace, ale pro získání chybové matice. Jednotlivé chybové matice byly sečteny a na závěr vyděleny ½ počtu všech učení neuronové sítě. Celý *cross-validation* test byl poté 3x zopakován. Chybové matice z jednotlivých *cross-validation* testů byly na závěr zprůměrovány. Z těchto důvodu, odpovídá součet prvků v chybové matice počtu shluků budov v ručně klasifikované množině. Výslednou chybovou matici lze vidět na obrázku č. 43 a metriky z ní vycházející poté v tabulce č. 16.



Obrázek 43: Chybová matice pro CNN
***Zdroj**: vlastní zpracování*

Celková přesnost klasifikace pomocí CNN dosáhla téměř 90 %. Nejlépe ze všech analyzovaných typů zástavby byl klasifikován typ zástavby *Samota* (*F-score = 0,989*). Všechny shluky budov tohoto typu zástavby byly klasifikátorem správně zařazeny (*Recall = 1*). Pouze několik málo shluků náležící typu zástavby *Liniová zástavba* bylo klasifikátorem označeno jako typ zástavby *Samota*. Druhým nejpřesněji klasifikovaným typem zástavby byly shluky budov náležící *Liniové zástavbě*. Hodnoty *F-score* přibližně 0,9 dosáhly typy zástavby *Blok* a *Industriální zástavba.* Oproti *Industriální zástavbě* byl počet shluků typu zástavby *Blok* pomocí klasifikátoru nadhodnocen. Nejhůře byly opět klasifikovány typy zástavby *Blok s vnitroblokem* a *Malé a střední domy*. Oba typy dosáhly hodnoty *F-score* přibližně 0,82. Relativně velké množství shluků budov typu *Blok s vnitroblokem* bylo klasifikátorem označeno jako *Malé a střední domy*. Z tohoto důvodu

dosáhl typ zástavby *Blok s vnitroblokem* vyšší hodnoty *Precision* než *Recall*. Pro typ zástavby *Malé a střední domy* pak platí přesný opak.

Typ zástavby	Precision	Recall	F-score
Blok	0,874	0,933	0,902
Blok s vnitroblokem	0,874	0,776	0,822
Malé a střední domy	0,792	0,864	0,827
Industriální zástavba	0,927	0,864	0,894
Liniová zástavba	0,933	0,933	0,933
Samota	0,978	1,000	0,989
		Accuracy	**0,895**

Tabulka 16: Analýza přesnosti klasifikace pomocí CNN
***Zdroj**: vlastní zpracování*

5 Srovnání klasifikátorů

Tato kapitola je věnována porovnání klasifikací zástavby pomocí tří klasifikátorů představených v kapitole 4.7. Jako první je analyzována přesnost klasifikace zástavby pouze pomocí ručně vybrané trénovací množiny. Druhá podkapitola je poté věnována zhodnocení klasifikace všech přibližně 170 tisíc shluků budov.

5.1 Porovnání klasifikátorů na ručně vybrané trénovací množině

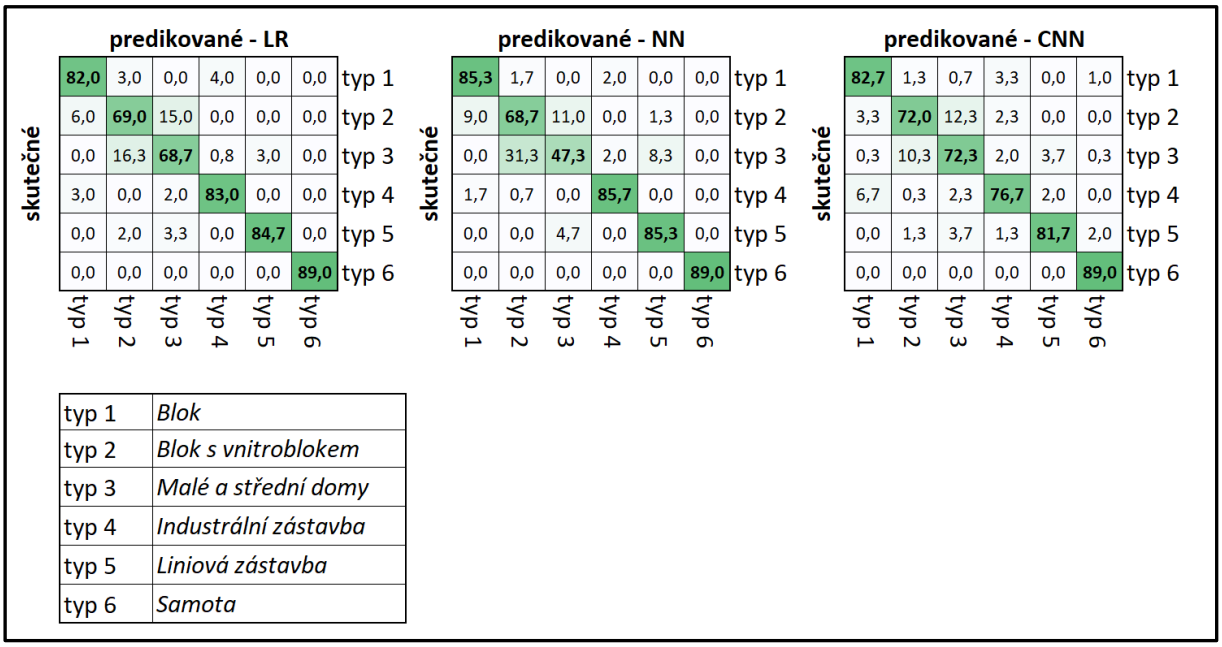
Cílem této podkapitoly je poskytnout bližší pohled na rozdíly v přesnosti klasifikace jednotlivých typů zástavby pomocí tří zmíněných klasifikátorů z kapitoly 4.7. Vzhledem k tomu, že pro klasifikátory založené na neuronových sítí se neukázalo jako přínosné využití klasifikovaných shluků budov pomocí logistické regrese, je toto porovnání založeno pouze na shlucích budov z ručně vybrané trénovací množiny obsahující přibližně 530 shluků budov (kapitola 4.7). Aby mělo porovnání co nejvyšší vypovídající schopnost, byly jednotlivé klasifikátory vždy trénovány a testovány na absolutně shodných shlucích budov.

Pro porovnání jednotlivých klasifikátorů byl použit stejný algoritmus jako pro zhodnocení přesnosti klasifikace zástavby pomocí CNN v kapitole 4.7.3.3. Algoritmus využívá *cross-validation* testu. Vstupní data jsou pomocí *cross-validation* testu rozdělena na 4 části (k=4). Jedna z částí je rozdělena na dvě další části určené pro validaci a testování (obrázek 39). Vzhledem k tomu, že klasifikátor logistické regrese během procesu fitování nijak nevyužívá validační část dat, je tato část spojena s trénovací. Celkem je každý klasifikátor v rámci jednoho *cross-validation* testu naučen a otestován 8x. Jednotlivé klasifikátory byly nastaveny pomocí stejných hodnot parametrů, které byly využity pro vyhodnocení přesnosti klasifikace jednotlivých klasifikátorů. Pro klasifikátor založený na CNN byly rastry náležící trénovací množině upraveny pomocí datové augmentace. Validační a testovací data byla poté využita v originální podobě.

Pro možné porovnání bylo pro každý klasifikátor v rámci *cross-validation* testu zaznamenáváno několik hodnot. Pomocí testovací části dat byla vždy spočtena přesnost klasifikace jako poměr správně klasifikovaných shluků budov ku všem shlukům budov v testovací části. Pomocí testovací části dat byla též sestrojena chybová matice. Pro každý klasifikátor byl také měřen čas od počátku učení/fitování klasifikátoru po ukončení testování. Všech osm chybových matic vzniklých v rámci jednoho *cross-validation* testu bylo sečteno. Celý *cross-validation* test byl pro zpřesnění výsledků 3x zopakován.

Chybové matice vzniklé z každého *cross-validation* testu byly na závěr zprůměrovány. Díky tomu odpovídá součet hodnot ve výsledné chybové matici pro každý klasifikátor součtu všech shluků v ručně vybrané trénovací množině. Podobně jako při hodnocení přesnosti klasifikace jednotlivých klasifikátorů byly z chybové matice spočteny hodnoty *precision, recall* a *F-score*.

Výsledné chybové matice pro každý z klasifikátorů lze vidět na obrázku č. 44. Hodnoty zmíněných metrik vycházejících z chybové matice jsou zaneseny v tabulce č. 17. V tabulce č. 18 jsou poté zaneseny hodnoty všech 24 měření klasifikační přesnosti a času procesu učení a testování.



Obrázek 44: Porovnání chybových matic
***Zdroj**: vlastní zpracování*

Hodnoty celkové přesnosti klasifikace (*accuracy*) pro jednotlivé klasifikace jsou velice podobné hodnotám získaným při analýze jednotlivých klasifikátorů. Drobné odchylky mohou být způsobeny tím, že klasifikátory v kapitole 5.1 mohly být učeny na jiných shlucích budov než v kapitole 4.7. Klasifikátory logistické regrese a CNN dosáhly celkové přesnosti klasifikace okolo 89 %. Pomocí neuronové sítě využívající popisných charakteristik byla přesnost klasifikace přibližně o 3 % nižší.

	precision			*recall*			*F-score*		
	LR	NN	CNN	LR	NN	CNN	LR	NN	CNN
Blok	0,901	0,889	0,889	0,921	0,959	0,929	0,911	0,923	0,908
Blok s vnitroblokem	0,764	0,671	0,844	0,767	0,763	0,800	0,765	0,714	0,821
Malé a střední domy	0,772	0,751	0,792	0,773	0,532	0,813	0,772	0,623	0,802
Industrální zástavba	0,945	0,955	0,895	0,943	0,973	0,871	0,944	0,964	0,883
Liniová zástavba	0,966	0,898	0,935	0,941	0,948	0,907	0,953	0,923	0,921
Samota	1,000	1,000	0,964	1,000	1,000	1,000	1,000	1,000	0,982

	LR	NN	CNN
accuracy	0,891	0,862	0,887

Tabulka 17: Porovnání klasifikátorů vycházející z chybových matic
***Zdroj**: vlastní zpracování*

Z tabulky č. 18 lze vyčíst relativně velký rozptyl mezi hodnotami přesnosti klasifikace při jednotlivých testováních. Vyšší rozdíly lze pozorovat u obou neuronových sítí spíše než u logistické regrese. Zejména CNN má rozdíl mezi nejvyšší a nejnižší naměřenou přesností klasifikace, a to více než 20 %. Část této heterogenity může být vysvětlena využitím náhodné veličiny při procesu učení neuronové sítě. Pro zmenšení rozdílů v dosažené přesnosti by bylo vhodné učení neuronové sítě na stejných trénovacích datech několikrát zopakovat a zaznamenat pouze nejvyšší naměřenou přesnost klasifikace během opakovaného učení. Vyšší rozdíly v dosažené přesnosti u neuronových sítí mohou být též způsobeny relativně malou trénovací množinou a tím pádem i větší závislostí mezi prvky použitými pro učení a testování a výslednou přesností klasifikace. Pro zjištění, zda vysoká hodnota přesnosti klasifikace dosažená jedním klasifikátorem implikuje vysokou hodnotu přesnosti klasifikace jiným klasifikátorem, byly spočteny korelační koeficienty mezi hodnotami přesnosti klasifikace pro jednotlivé klasifikátory při jednotlivých měření. Matici korelačních koeficientů lze vidět v tabulce č. 19. Pro logistickou regresi a neuronovou sít byla naměřena hodnota korelačního koeficientu, kterou lze podle (Evans 1996) označit jako středně silnou. Určitá závislost byla prokázána též mezi hodnotami přesnosti klasifikace pomocí neuronové sítě a CNN. Naopak hodnoty přesnosti klasifikace pomocí logistické regrese a CNN spolu korelují velice slabě. Dobře to lze vidět například u 3. a 17. měření v tabulce č. 18. Z tabulky č. 18 lze též vyčíst výrazný rozdíl nutný pro naučení a otestování jednotlivých klasifikátorů. Tento proces je v případě neuronových sítě přibližně 500x delší než v případě logistické regrese. Pro CNN trvá tento proces ještě 10x déle. Rozdíly v naměřeném čase v případě neuronových sítí jsou způsobeny zejména proměnlivým počtem epoch nutných pro

naučení sítě, neboť bylo stejně jako v kapitolách 4.7.2 a 4.7.3 využito funkce *EarlyStopping*.

Z výsledných chybových matic (obrázek 44) a metrik z nich vycházejících (tabulka 17) lze pro všechny klasifikátory vypozorovat podobný vzorec. Nejvyšší přesnosti klasifikace je vždy dosaženo pro typ zástavby *Samota*. Relativně vysoká hodnota *F-score* je naměřena u typů zástavby: *Blok, Industriální zástavba* a *Liniová zástavba*. Nejnižší přesnost klasifikace je vždy naměřena u typů *Blok s vnitroblokem* a *Malé a střední domy*, u kterých dochází k relativně časté záměně.

	přesnost			čas [s]		
	LR	NN	CNN	LR	NN	CNN
1	0,875	0,905	0,901	0,036	20,115	169,480
2	0,920	0,845	0,947	0,034	13,813	193,990
3	0,981	0,920	0,870	0,039	23,535	197,908
4	0,905	0,936	0,947	0,028	16,195	217,737
5	0,860	0,800	0,916	0,035	14,010	114,765
6	0,860	0,830	0,855	0,028	11,770	165,728
7	0,860	0,858	0,868	0,025	13,620	234,289
8	0,858	0,770	0,825	0,033	13,430	224,883
9	0,875	0,920	0,962	0,036	21,249	268,134
10	0,920	0,860	0,947	0,033	18,557	196,954
11	0,981	0,890	0,809	0,033	21,179	189,869
12	0,905	0,936	0,840	0,028	27,541	105,721
13	0,860	0,815	0,916	0,035	19,665	221,080
14	0,860	0,815	0,779	0,030	15,959	216,961
15	0,860	0,873	0,930	0,027	16,266	163,545
16	0,858	0,770	0,809	0,033	18,870	218,583
17	0,875	0,890	0,992	0,038	18,950	145,971
18	0,920	0,875	0,962	0,036	22,394	248,298
19	0,981	0,890	0,886	0,033	17,573	176,831
20	0,905	0,951	0,901	0,028	15,517	234,113
21	0,860	0,800	0,870	0,036	9,659	170,364
22	0,860	0,785	0,840	0,040	16,790	104,899
23	0,860	0,873	0,868	0,032	19,133	193,653
24	0,858	0,815	0,855	0,033	28,822	276,313

Tabulka 18: Hodnoty přesnosti klasifikace a času učení + testování
***Zdroj**: vlastní zpracování*

Typ zástavby *Blok* nejlépe dokázala klasifikovat neuronová síť, ostatní dva klasifikátory však dosáhly pouze o 0,01 nižší hodnoty *F-score*. Typy zástavby *Blok s vnitroblokem* a *Malé a střední domy* nejlépe dokázala klasifikovat konvoluční neuronová síť. Logistická regrese v tomto případě dosáhla o přibližně 0,05 nižší hodnoty *F-score*, neuronová síť dokonce o více než 0,1. Naopak typ *Industriální zástavba* dokázala neuronová síť klasifikovat nejlépe a CNN dosáhla o téměř 0,08 nižší hodnoty *F-score*. Liniovou zástavbu nejlépe dokázala klasifikovat logistická regrese, avšak obě neuronové sítě dosáhly pouze o 0,03 nižší hodnoty *F-score*. Typ zástavby *Samota* dokázaly

klasifikovat všechny tři klasifikátory velmi dobře. Logistická regrese a neuronová síť dosáhly dokonce hodnoty *F-score* = 1. CNN dosáhla hodnoty přibližně 0,98.

	LR	*NN*	*CNN*
LR	1		
NN	0,547	1	
CNN	0,083	0,414	1

Tabulka 19: Matice korelačních koeficientů
***Zdroj**: vlastní zpracování*

Jak již bylo zmíněno dříve, nejméně přesně byly klasifikovány typy *Blok s vnitroblokem* a *Malé a střední domy*. Vzhledem k tomu, že byly tyto typy zástavby nejhůře klasifikovány jak klasifikátory založené na popisných charakteristikách, tak na vizuálním vjemu, může být problém i v navržené typologii zástavby, kdy mezi těmito typy zástavby není přesně daná hranice. Nižší přesnost klasifikace těchto dvou typů zástavby může být též způsobena nevhodným zařazením některých shluků budov do ručně vybrané trénovací množiny. Zbylé typy zástavby se však, v případě ručně vybrané trénovací množiny, již podařilo klasifikovat relativně dobře.

5.2 Porovnání klasifikací všech shluků budov

Cílem této podkapitoly je porovnat klasifikace všech přibližně 170 tisíc shluků pomocí tři klasifikátorů popsaných v kapitole 4.7. První část této podkapitoly není věnována přímo analýze přesnosti klasifikace všech 170 tisíc shluků budov, ale spíše charakteristice jednotlivých klasifikací. Druhá část je poté věnována právě analýze odhadu přesnosti klasifikace pomocí porovnání klasifikovaného typu zástavby a kartografické reprezentace daného shluku budov na ZM 50.

5.2.1 Charakteristika klasifikace všech shluků budov

Před vlastní charakteristikou jednotlivých klasifikací bylo nutné všech 170 tisíc shluků budov klasifikovat pomocí již zmíněných třech klasifikátorů. Pro každý klasifikátor bylo, kromě získání nejpravděpodobnějšího typu zástavby daného shluku budov, využito možnosti určení pravděpodobností zařazení daného shluku budov k jednotlivým typům zástavby. Stejného způsobu bylo využito při analýze možného využití shluků budov, klasifikovaných pomocí logistické regrese, pro rozšíření trénovací množiny neuronových sítí.

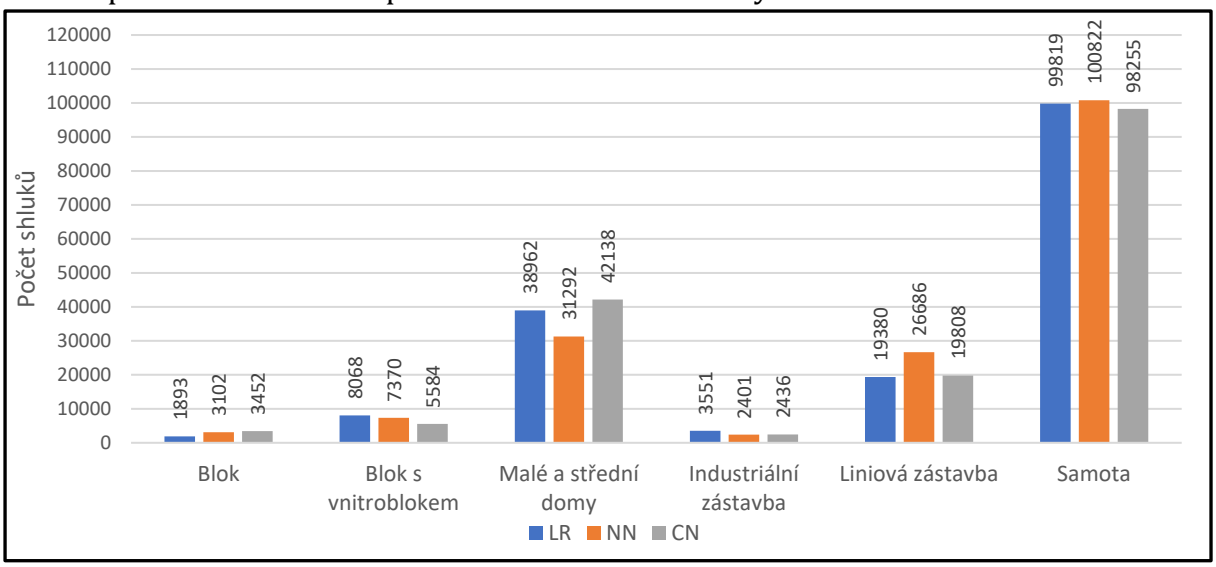
Jako první byly všechny shluky budov klasifikovány pomocí klasifikátoru logistické regrese. Predikce pomocí tohoto klasifikátoru je nejjednodušší a zároveň nejrychlejší ze všech tří klasifikátorů. Výhodou tohoto klasifikátoru je fakt, že při použití stejných

trénovacích dat je výsledný klasifikátor natrénován vždy stejně. Zároveň pro tento klasifikátor není nutné vymezit validační část dat pro potřebu fitování. Pro naučení tohoto klasifikátoru bylo pro účel klasifikace všech shluků budov využito veškerých shluků budov v ručně vybrané trénovací množině. Samotná klasifikace všech 170 000 shluků trvala pouze několik sekund.

V případě neuronových sítí (včetně CNN) již neplatí, že při použití stejných trénovacích dat je výsledná síť naučena pokaždé stejně. Zároveň je vhodné, vymezit validační část dat, která nepřímo pomocí funkce *EarlyStopping* ovlivňuje, jak bude síť naučena. Z těchto vlastností neuronových sítí vyvstávají dvě otázky. Jak velkou část vstupních dat vymezit jako validační? Jak posoudit, že jedna neuronová síť je naučena lépe než jiná?

Jedním z nejčastěji používaných rozdělení vstupních dat na trénovací a validační část je poměr 80:20 vycházející z takzvaného Paretova pravidla (Macek 2008). Tento poměr je však vhodný pro relativně velké datasety (Cawley, Talbot 2010). Vzhledem k tomu, že ručně vybraná trénovací množina obsahuje pouze několik stovek prvků, byl pro účely této práce zvolen poměr rozdělení vstupních dat na trénovací a validační část 70:30. Pro výběr nejlépe naučené neuronové sítě bylo využito hodnoty *Validation Loss*, neboli hodnoty ztrátové funkce vypočítané z validační části dat v okamžiku, kdy je proces učení sítě ukončen pomocí funkce *EarlyStopping*. Proces učení byl v případě neuronové sítě, využívající popisné charakteristiky shluků budov, zopakován celkem 50x. Před každým učením jsou data vždy náhodně rozdělena na trénovací a validační část. Podmínkou je však zachování stejného rozložení počtu shluků budov náležícím jednotlivým typům. Za nejvhodnější byla poté vybrána ta, která dosáhla nejnižší hodnoty *Validation loss*. Čím nižší tato hodnota je, tím je větší předpoklad, že bude daná síť lépe generalizovat, jinými slovy lépe reagovat na neznámá data. Doba predikce všech 170 000 shluků budov byla jen nepatrně delší než v případě logistické regrese. Pro CNN byl proces učení, vzhledem k mnohem větší časové náročnosti, zopakován pouze 30x. Doba predikce všech rastrových podob shluků budov byla již mnohonásobně delší. V prostředí *Google Colab* trvala predikce všech 170 000 rastrů přibližně jednu hodinu.

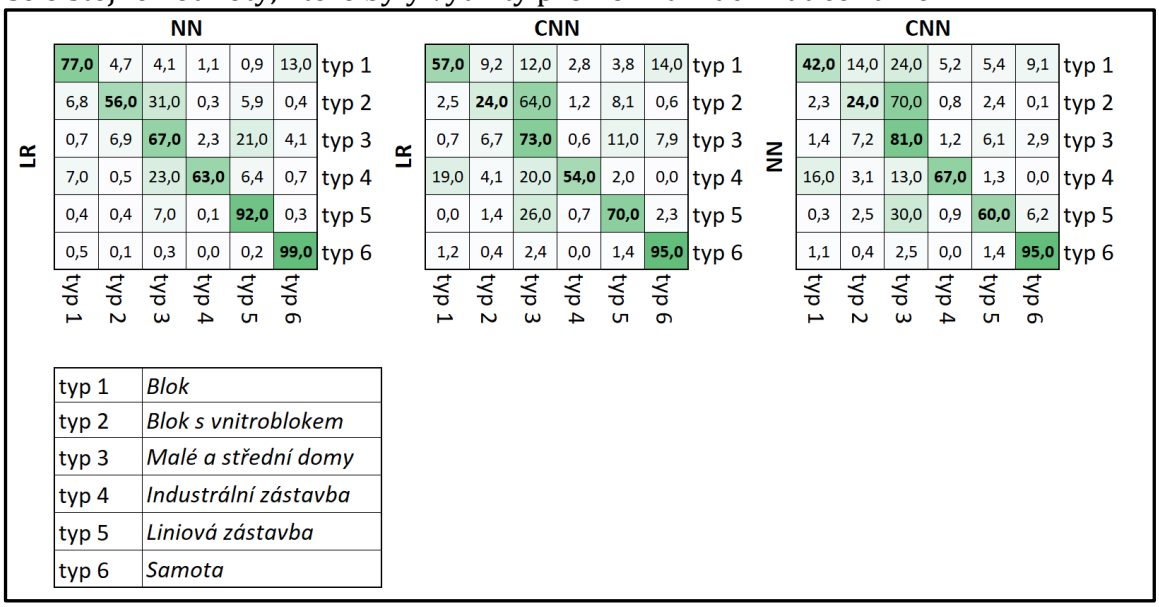
Na grafu č. 11 lze vidět rozložení počtu shluků budov náležící jednotlivým typům zástavby podle jednotlivých klasifikátorů. Téměř 60 % všech shluků budov bylo všemi třemi klasifikátory označeno jako typ zástavby *Samota.* V tomto typu zástavby jsou též nejmenší rozdíly v počtu shluků budov mezi jednotlivými klasifikátory. To odpovídá předchozím analýzám přesnosti klasifikace, kdy byl typ *Samota* vždy klasifikován nejpřesněji. Druhým nejvíce zastoupeným typem zástavby jsou *Malé a střední domy*. Počty shluků budov tohoto typu se mezi jednotlivými klasifikátory již výrazně liší. CNN přiřadila tomuto typu o 10 000 víc shluků než neuronová síť. To představuje 30% rozdíl. Více než 10 % všech shluků bylo poté označeno jako *Liniová zástavba.* V tomto případě naopak počet shluků budov určený neuronovou sítí jako *Liniová zástavba* výrazně převyšuje počet shluků budov určený zbývajícími klasifikátory. Již pouze okolo 4 % všech shluků bylo označeno jako typ *Blok s vnitroblokem*. Tomuto typu zástavby přiřadil klasifikátor logistické regrese téměř o 1/3 více shluků budov než CNN. Velice podobná situace nastala u typu *Industriální zástavba*. Naopak u posledního typu zástavby *Blok* je situace téměř opačná. Výraznější rozdíly v počtu shluků mezi jednotlivými klasifikátory mohou poukazovat na nižší přesnost klasifikace některým z klasifikátorů.



Graf 11: Histogram četností shluků v jednotlivých typech zástavby
Zdroj: *vlastní zpracování*

Další srovnání jednotlivých klasifikací všech shluků budov bylo provedeno pomocí sestrojení chybových matic (obrázek 45). Na rozdíl od předchozího použití chybové matice, kdy byly srovnávány skutečné a predikované hodnoty, jsou v tomto případě srovnávány predikované hodnoty ze dvou klasifikátorů. Z tohoto důvodu je pro tyto matice vhodnější užívat název matice záměn, spíše než chybová matice. Vzhledem k tomu, že počty shluků budov náležící jednotlivým typům zástavby jsou značně nevyrovnané,

jsou matice záměn pro větší přehlednost znázorněny v normalizované podobě. Při normalizaci jsou hodnoty v řádku matice vyděleny součtem hodnot celého řádku. Hodnoty na diagonále poté odpovídají hodnotě *Recall*. V případě, že by pro normalizaci byl využit součet hodnot ve sloupci, hodnoty na diagonále by poté odpovídaly hodnotě *Precision*. Stejně jako v předchozích případech hodnocení přesnosti klasifikace pomocí chybových matic, jsou i v tomto případě spočteny hodnoty metrik vycházejících z jednotlivých matic záměn (tabulka 20). Vzhledem k tomu, že počty shluků v jednotlivých klasifikovaných typech zástavby jsou značně nevyvážené, je tabulka č. 20. doplněna o vážený průměr hodnoty tří metrik pro každou kombinaci klasifikátorů. Vahami jsou v tomto případě počty shluků v jednotlivých klasifikovaných typech. Jedná se o stejné hodnoty, které byly využity pro normalizaci matice záměn.



Obrázek 45: Matice záměn mezi jednotlivými klasifikátory
***Zdroj**: vlastní zpracování*

Nejvyšší shoda klasifikací byla naměřena mezi logistickou regresí a neuronovou sítí. Oba tyto klasifikátory klasifikovaly shodně celkem 88 % všech shluků budov. Tato na první pohled vysoká shoda je dána zejména téměř 100% shodou v případě typu *Samota* a též výrazně vyšším počtem shluků budov v tomto typu. V případě, že by všechny typy zástavby byly reprezentovány stejným počtem shluků budov, dosáhla by shodnost klasifikací přibližně 75 %. Jak lze z tabulky č. 20. vyčíst, nejmenší shoda mezi logistickou regresí a neuronovou sítí nastala v případě typů zástavby *Blok* a *Blok s vnitroblokem*. Pouze 47 % ze shluků budov označených neuronovou sítí jako typ *Blok* bylo logistickou regresí označeno jako stejný typ a pouze 56 % ze shluků budov označených logistickou regresí jako *Blok s vnitroblokem* bylo označeno neuronovou sítí jako stejný typ zástavby.

Z chybových matic lze též vyčíst, že pomocí klasifikátoru neuronových sítí byl, oproti klasifikátoru logistické regrese, nadhodnocen počet shluků budov u typů *Malé a střední domy* a *Liniová zástavba,* a naopak podhodnocen počet shluků budov u klasifikovaných typů *Blok s vnitroblokem* a *Industriální zástavba*.

	precision			***recall***			***F-score***		
	LR/NN	LR/CNN	NN/CNN	LR/NN	LR/CNN	NN/CNN	LR/NN	LR/CNN	NN/CNN
Blok	0,47	0,32	0,38	0,77	0,57	0,42	0,58	0,41	0,40
Blok s vnitroblokem	0,61	0,35	0,32	0,56	0,24	0,24	0,58	0,28	0,28
Malé a střední domy	0,84	0,68	0,60	0,67	0,73	0,81	0,75	0,71	0,69
Industrální zástavba	0,93	0,79	0,66	0,63	0,54	0,67	0,75	0,65	0,67
Liniová zástavba	0,67	0,68	0,81	0,92	0,70	0,60	0,77	0,69	0,69
Samota	0,98	0,96	0,97	0,99	0,95	0,95	0,99	0,95	0,96
průměr	0,75	0,63	0,62	0,76	0,62	0,62	0,74	0,62	0,62
vážený průměr	0,89	0,83	0,84	0,88	0,82	0,83	0,88	0,82	0,83

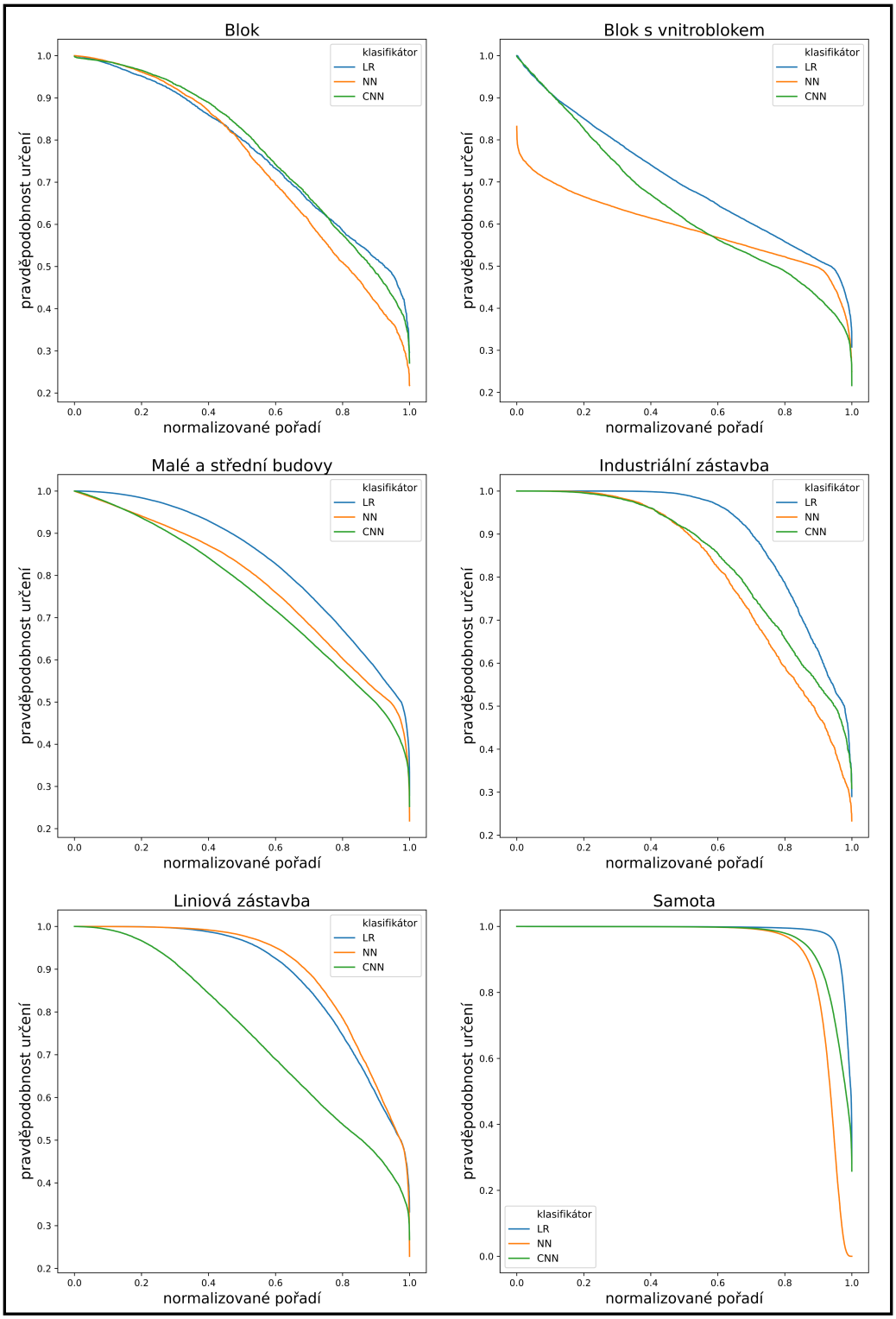
celková shoda

LR/NN	LR/CNN	NN/CNN
0,88	0,82	0,83

Tabulka 20: Metriky vycházející z matice záměn jednotlivých klasifikátorů
***Zdroj**: vlastní zpracování*

Shoda mezi klasifikátorem logistické regrese a klasifikátorem CNN je již výrazně nižší. Celková shoda dosáhla v tomto případě hodnoty přibližně 82 %, ale stejně jako u shody klasifikací mezi logistickou regresí a neuronovou sítí je toto číslo výrazně navýšeno vysokou shodou v případě klasifikovaného typu *Samota*. Nejmenší shoda těchto dvou klasifikací byla naměřena v případě typu zástavby *Blok s vnitroblokem*. Pouze 24 % ze shluků budov označených klasifikátorem logistické regrese jako typ *Blok s vnitroblokem*, bylo pomocí CNN označeno jako stejný typ zástavby. Z opačného úhlu pohledu dosáhla shoda pouze 32 %.

Celková shoda mezi klasifikací zástavby pomocí neuronové sítě a pomocí CNN dosáhla hodnoty 83 %. Obdobně jako u dvou předchozích porovnání klasifikátorů, je tato hodnota navýšena zejména vysokou shodou v případě typu zástavby *Samota*. Stejně jako v případě porovnání klasifikace pomocí CNN s klasifikací pomocí logistické regrese, i v tomto případě byla nejmenší shoda naměřena u klasifikovaného typu *Blok s vnitroblokem*. Obecně lze o klasifikátoru CNN říct, že oproti klasifikátorům využívající popisných charakteristik, výrazně podhodnocuje počet shluků budov v klasifikovaném typu *Blok s vnitroblokem,* a naopak výrazně nadhodnocuje počet shluků budov v typu zástavby *Malé a střední domy*.



Graf 12: Porovnání pravděpodobnosti určení shluků budov
Zdroj: *vlastní zpracování*

Pro další srovnání jednotlivých klasifikací bylo využito hodnot pravděpodobnosti přiřazení daného shluku k jednotlivým typům zástavby pomocí daného klasifikátoru. Cílem tohoto kroku je zjistit, s jakou pravděpodobností jsou shluky budov, náležící jednotlivým typům zástavby, určovány a jak se tato vlastnost liší pro jednotlivé klasifikátory. Pro sestavení požadovaných grafů jsou nejprve shluky budov náležící

danému typu zástavby seřazeny sestupně podle hodnoty pravděpodobnosti určení. Pro možné porovnání jednotlivých klasifikátorů je na osu X vynášeno normalizované pořadí daného shluku. Na ose Y poté příslušná hodnota pravděpodobnosti určení (graf 12).

Ze všech grafů vyplývá, že mezi jednotlivými klasifikátory nejsou příliš velké rozdíly. Jinak řečeno, rozložení pravděpodobnosti určení shluků budov daného typu zástavby je pro všechny klasifikátory relativně podobné. Největší rozdíl je patrný u typu *Liniová zástavba*. Shluky budov tohoto typu zástavby jsou pomocí CNN klasifikovány s menší pravděpodobností, neboť klasifikátor CNN nepřiřazuje shlukům budov tento typ s takovou jistotou, jako dva zbývající. Tento jev může souviset s tím, že CNN dosahují u tohoto typu nižší klasifikační přesnosti než zbývající dva klasifikátory. Podobný případ lze pozorovat i v případě typu *Blok s vnitroblokem*, kde neuronová síť určuje shluky budov tohoto typu s nižší jistotou než zbývající dva klasifikátory.

Větší rozdíly než mezi klasifikátory lze pozorovat mezi jednotlivými typy zástavby. S nejvyššími hodnotami pravděpodobnosti příslušnosti k danému typu jsou klasifikovány shluky budov typu *Samota*. Více než 75 % všech shluků budov označených jako typ *Samota* bylo určeno s pravděpodobností vyšší než 95 %. Zároveň byl tento typ zástavby klasifikován vždy s nejvyšší přesností. Relativně s vysokou pravděpodobností určení byly klasifikovány shluky budov označené jako typ *Liniová zástavba* a *Industriální zástavba*. O obou těchto typech zástavby lze říct, že byly klasifikovány s relativně vysokou přesností. Z grafu též vyplývá, že i přes nižší přesnost klasifikace, byly shluky budov, náležící typu *Malé a střední domy,* klasifikovány s vyšší pravděpodobností určení než shluky budov typu zástavby *Blok*. S nejmenší jistotou poté byly klasifikovány shluky budov náležící typu *Blok s vnitroblokem*. Nejspíše i u tohoto typu souvisí nižší pravděpodobnost určení s nižší naměřenou přesností klasifikace.

5.2.2 Analýza přesnosti klasifikace všech shluků budov

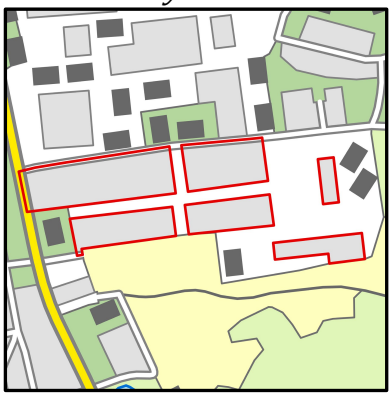
Cílem této podkapitoly práce je pokusit se zhodnotit přesnost klasifikace všech přibližně 170 000 shluků budov. Referenčními daty jsou v tomto případě generalizované polygony zástavby na ZM 50, respektive jejich rozdílné kartografické reprezentace. Vzhledem k tomu, že typologie zástavby představená v kapitole 3 byla tvořena i s ohledem na rozdílnou kartografickou generalizaci jednotlivých typů, je pro každý typ zástavby známá i očekávaná kartografická reprezentace na ZM 50. V první části této podkapitoly je představena typologie kartografických reprezentací zástavby na ZM 50.

Následně je popsán algoritmus pro přiřazení odpovídající kartografické reprezentace každému shluku budov. Na závěr je pak provedeno samotné zhodnocení přesnosti klasifikace.

Obrázek 46: Kategorie zástavby na ZM 50: Bloková zástavba
Zdroj: ČÚZK (2020a), *vlastní zpracování*

5.2.2.1 Typologie kartografických reprezentací zástavby na ZM 50

Celkem byly pro účely této diplomové práce vymezeny 4 kategorie zástavby vycházející ze třech různých kartografických reprezentací zástavby na ZM 50. První vymezenou kategorií je **Bloková zástavba** (obrázek 46). Tato kategorie je na ZM 50 reprezentována světle šedými polygony, které kompletně vyplňují meziuliční prostor. Jinak řečeno, hranice těchto polygonů je tvořena uliční sítí. Z pohledu kartografické generalizace se jedná o příklad agregace. Nejčastěji je touto kartografickou reprezentací znázorňována zástavba v centrech měst, popřípadě hustěji umístěné rodinné domy. Na obrázku č. 46 je znázorněno několik bloků této kategorie. Jejich hranice je zvýrazněna pomocí červené linie. Touto kartografickou reprezentací jsou znázorňovány zejména klasifikované typy zástavby *Blok, Blok s vnitroblokem* a v některých případech i *Malé a střední domy.*

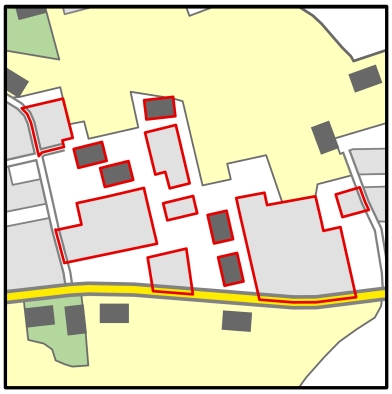


Obrázek 47: Kategorie zástavby na ZM 50: Neohraničený blok
Zdroj: ČÚZK (2020a), *vlastní zpracování*

Druhou vymezenou kategorií znázornění zástavby na ZM 50 je **Neohraničený blok** (obrázek 47). Obdobně jako první zmíněná kategorie je i tato kategorie znázorněna na ZM 50 pomocí světle šedého polygonu. Oproti předchozí kategorii nejsou polygony kategorie kompletně ohraničeny uliční či silniční sítí. V tomto případě se z pohledu kartografické generalizace jedná zejména o zjednodušení, případně agregaci. Touto kartografickou reprezentací jsou na ZM 50 znázorňovány zejména velké budovy jako jsou například průmyslové haly. Touto kategorií zástavby na ZM 50 mohou být též znázorněny klasifikované typy zástavby zmíněné u první kategorie **Bloková zástavba**, které nejsou kompletně obklopeny uliční sítí. Na obrázku č. 47 je opět ukázka několika příkladů kategorie **Neohraničený blok**. Výskyt této kategorie zástavby na ZM 50 je očekávám zejména v případě klasifikovaného typu *Industriální zástavba*, ale též u klasifikovaných typů *Malé a střední domy, Blok* a *Blok s vnitroblokem*.

Obrázek 48: Kategorie zástavby na ZM 50: Samostatná budova
Zdroj: ČÚZK (2020a), *vlastní zpracování*

Následující kategorií znázornění zástavby na ZM 50 je **Samostatná budova** (obrázek 48). Tato kategorie je na ZM 50 znázorněna pomocí bodového symbolu, který je představován tmavě šedým obdélníkem. Velikost tohoto obdélníku je vždy stejná, mění se pouze jeho orientace. Tato kategorie je z hlediska kartografické generalizace nejčastěji příkladem typifikace. Tímto bodovým znakem jsou na ZM 50 znázorňovány zejména samostatné malé budovy, respektive skupiny například rodinných domů s většími rozestupy. Několik příkladů této kategorie znázornění zástavby na ZM 50 lze vidět na obrázku č. 48. Z klasifikovaných typů je výskyt této kartografické reprezentace zástavby očekáván zejména u typu *Samota*. Výskyt se dá též předpokládat u typu *Liniová zástavba* a *Malé a střední domy*.



Obrázek 49: Kategorie zástavby na ZM 50: Kombinovaná kategorie
Zdroj: ČÚZK (2020a), *vlastní zpracování*

Poslední kategorie byla nazvána jako **Kombinovaná kategorie** (obrázek 49). Jedná se totiž o kombinaci kartografických reprezentací zástavby na ZM 50 kategorií **Neohraničený blok** a **Samostatná budova**. Cílem této kombinace je vytvoření kategorie zástavby, která bude lépe vystihovat klasifikovaný typ *Industriální zástavba*. V tomto klasifikovaném typu se velice často mísí velké a malé budovy. Částečný výskyt této kategorie zástavby lze též očekávat u klasifikovaných typů zástavby *Malé a střední domy* a *Liniová zástavba*. U těchto dvou klasifikovaných typů lze tuto kategorii zástavby očekávat v místech s rozličnou hustotou zástavby uvnitř jednoho shluku budov. Příklad této kombinovaného kategorie lze vidět na obrázku č. 49.

5.2.2.2 Algoritmus

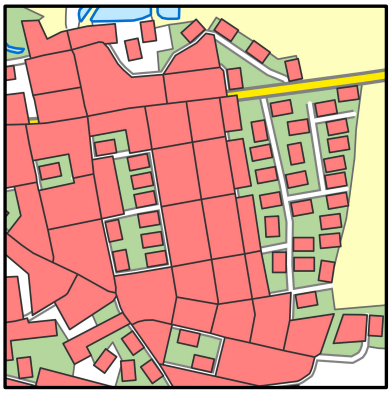
V této podkapitole je představen algoritmus, který nejprve identifikuje představené kategorie zástavby na ZM 50 a následně přiřadí každému shluku budov jemu příslušící kategorii zástavby na ZM 50. Pro identifikaci kategorií zástavby byly použity již zmíněné dvě vrstvy z volně dostupného datového zdroje *Data50*. Ukázku obou vrstev lze vidět na obrázku č. 50.

1. **Identifikace představených kategorií zástavby na ZM 50**

   Prvním krokem bylo upravení vstupních vrstev do podoby, jak je zástavba znázorněna na ZM 50. Vzhledem k tomu, že bodový znak zástavby na ZM 50 je ve vstupní vrstvě reprezentován pouze jako bodová vrstva, bylo jí nutné pro účely této práce převést na polygon. Pro každý pod je kromě jeho polohy známa informace ohledně rotace bodového znaku na ZM 50. Pomocí těchto údajů byl pro každý pod vytvořen polygon o velikosti 60 m x 40 m, který odpovídá kartografické reprezentaci zmíněného bodového znaku zástavby.

Obrázek 50: Vrstvy zástavby z datového zdroje Data50
Zdroj: ČÚZK (2020a), *vlastní zpracování*

Druhým krokem bylo rozdělení polygonu nacházejících se ve druhé vrstvě (obrázek 50, červeně) na menší polygony odpovídající kartografické reprezentaci kategorií zástavby na ZM 50 **Bloková zástavba** a **Neohraničený blok**. Pro proces rozdělení byl opět využit algoritmus představený v kapitole 4.3.1 pro segmentaci budov do shluků. Nejdříve jsou hranice polygonů převedeny na linii. Následně jsou tyto linie spojeny s vrstvou liniových bariér. Vrstvu liniových bariér stejně jako v případě segmentace budov do shluků tvoří spojená vrstva liniových prvků zmíněných v kapitole 4.1. Poslední částí dělení polygonů je převedení spojené liniové vrstvy hranic polygonu a liniových bariér zpět na polygony.



Obrázek 51: Rozdělená polygonová vrstva zástavby na ZM 50
Zdroj: ČÚZK (2020a), *vlastní zpracování*

Následujícím krokem je spojení polygonů vzniklých z bodové vrstvy a rozdělených polygonů do jediné společné vrstvy. Ukázku této spojené vrstvy lze vidět na obrázku č. 51. Polygonům, které byly vytvořeny z bodové vrstvy je přiřazena hodnota atributu *ZM_typ* = 3. Posledním krokem identifikace představených kategorií zástavby na ZM je odlišení těch polygonů, které jsou kompletně ohraničeny pomocí linií v dělící vrstvě. Pro tento účel byla vrstva liniových bariér převedena na polygony. V případě, že polygon reprezentující zástavbu a polygon vzniklý z vrstvy liniových bariér je totožný, je polygonu reprezentujícímu zástavbu na ZM 50 přiřazena hodnota atributu

ZM_typ = 1. Pokud si dané polygony neodpovídají, je polygonu reprezentující zástavbu přidělena hodnota atributu *ZM_typ* = 2. **Kombinovaná kategorie** zástavby není v této fázi algoritmu identifikována.

2. **Přiřazení kategorie zástavby na ZM 50 shlukům budov**

Vzhledem k tomu, že velice často je shluk budov překryt více polygony reprezentujícími různé kategorie zástavby na ZM 50, nebylo možné využít prostého prostorového přiřazení (prostorový JOIN). Z tohoto důvodu byl vytvořen algoritmus, který shlukům budov přiřadí kategorii zástavby na ZM 50 v závislosti na procentuálním zastoupení jednotlivých kartografických reprezentací na ZM 50 uvnitř daného shluku budov. Prvním krokem této operace je protnutí polygonové vrstvy shluků budov a polygonové vrstvy reprezentující zástavbu na ZM 50. Pomocí nástroje kontingenčních tabulek je poté spočítáno procentuální zastoupení jednotlivých kartografických reprezentací zástavby na ZM 50 uvnitř každého shluku. Následně byl vytvořen seznam 10 rozhodujících pravidel (obrázek 52), které v závislosti na různém procentuálním zastoupení přiřadí danému shluku budov jednu ze 4 kategorií zástavby na ZM 50 představených v kapitole 5.2.2.1. Soubor pravidel vznikl z řady experimentů na datech ručně vybrané trénovací množiny. Přesnost přiřazení kategorie zástavby na ZM 50 jednotlivým shlukům nebyla kvantifikována, avšak pro účely odhadu přesnosti klasifikace zástavby je přesnost přiřazení dostatečná.

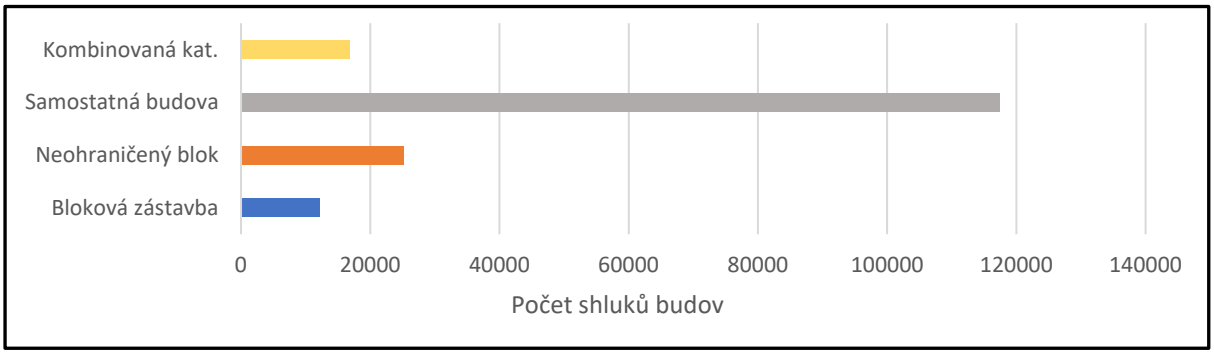
```
# ZM_typ_1 = procentuální zastoupení polygonu s hodnotou atributu ZM_typ = 1
# ZM_typ_2 = procentuální zastoupení polygonu s hodnotou atributu ZM_typ = 2
# ZM_typ_3 = procentuální zastoupení polygonu s hodnotou atributu ZM_typ = 3

if ZM_typ_1 == 0 and ZM_typ_2 == 0 and ZM_typ_3 == 0:
    Shluk_ZM_typ = samostatná budova
elif ZM_typ_1 > 50:
    Shluk_ZM_typ = bloková zástavba
elif ZM_typ_2 > 40 and ZM_typ_3 < 2:
    Shluk_ZM_typ = neohraničený blok
elif ZM_typ_2 > 0 and ZM_typ_3 == 0:
    Shluk_ZM_typ = neohraničený blok
elif ZM_typ_3 > 40 and ZM_typ_2 < 2:
    Shluk_ZM_typ = samostatná budova
elif ZM_typ_3 > 0 and ZM_typ_2 == 0:
    Shluk_ZM_typ = samostatná budova
elif ZM_typ_2 > 0 and ZM_typ_1 == 0 and ZM_typ_3 == 0:
    Shluk_ZM_typ = neohraničený blok
elif ZM_typ_3 > 0 and ZM_typ_1 == 0 and ZM_typ_2 == 0:
    Shluk_ZM_typ = samostatná budova
elif ZM_typ_1 > 0 and ZM_typ_2 == 0 and ZM_typ_3 == 0:
    Shluk_ZM_typ = samostatná budova
else:
    Shluk_ZM_typ = kombinovaná kategorie
```

Obrázek 52: Pravidla pro převod kategorií zástavby na ZM 50 shlukům budov
Zdroj: *vlastní zpracování*

5.2.2.3 Vlastní analýza přesnosti klasifikace

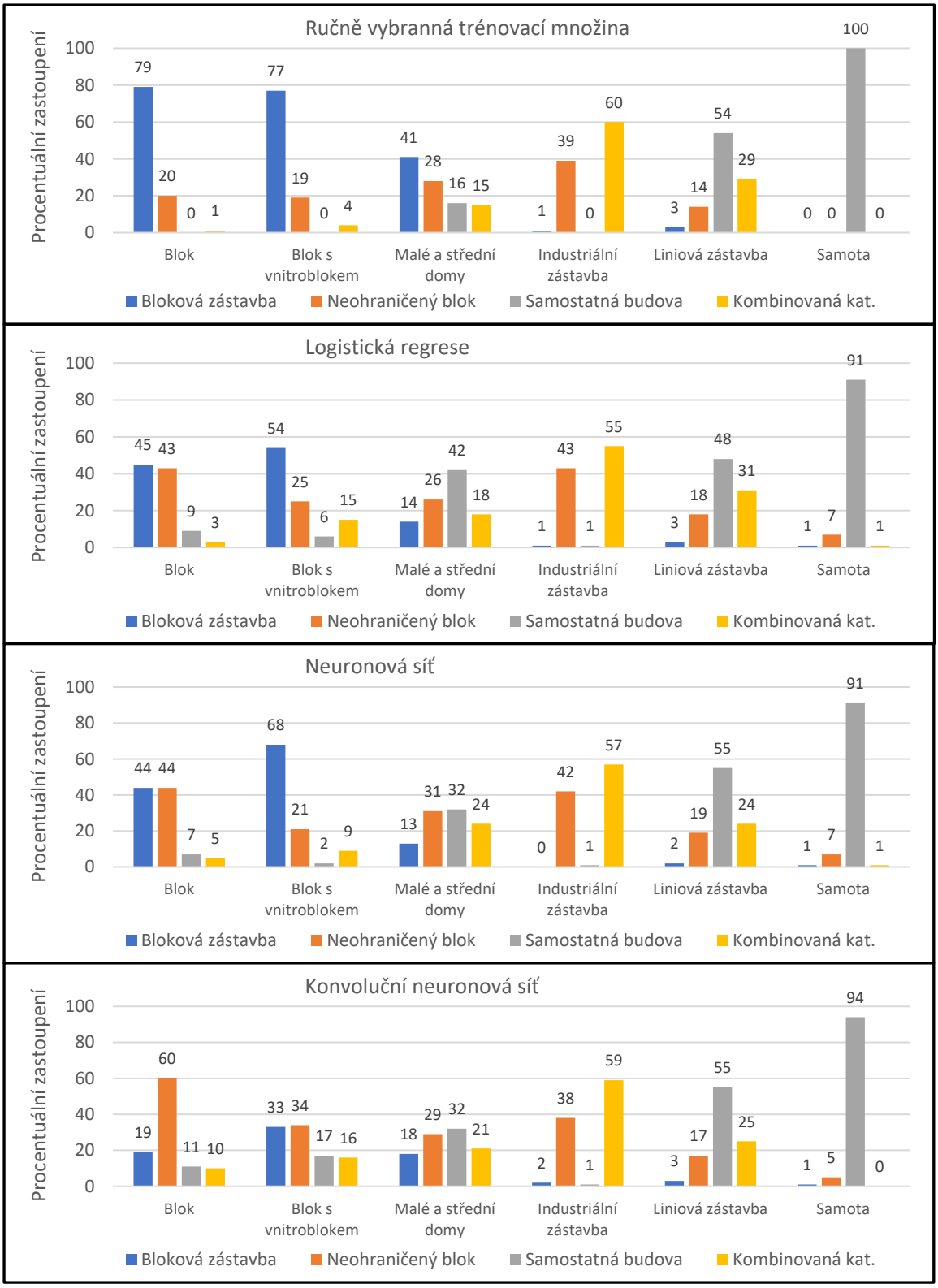
Před samotným srovnáním byl vytvořen histogram četností jednotlivých kategorií zástavby na ZM 50 u všech shluků budov (graf 13). Z tohoto grafu vyplývá velká převaha kategorie *Samostatné budovy*. Takto vysoký výskyt potvrzuje i vysoký počet shluků náležící do klasifikovaného typy *Samota*. Druhou nejčastěji se vyskytovanou kategorií zástavby na ZM 50 je *Neohraničený blok* následovaný *Kombinovanou* kategorií. Nejmenšími počtu shluků budov byla poté přiřazena kategorie *Bloková zástavba*.



Graf 13: Rozložení počtu shluků budov podle kategorie zástavby na ZM 50
***Zdroj:** vlastní zpracování*

Pro vlastní analýzu přesnosti klasifikace byly vytvořeny celkem čtyři histogramy. První histogram slouží jako referenční, neboť pro jeho sestrojení byla využita data pouze z ručně vybrané trénovací množiny. Zbylé histogramy poté vycházejí z klasifikací tří klasifikátorů. Cílem histogramů je graficky znázornit závislost mezi kategorií zástavby na ZM 50 a klasifikovaným typem. Vzhledem k velké disproporci v počtech shluků mezi jednotlivými klasifikovanými typy, byly hodnoty četností normalizovány.

Hodnoty z histogramu ručně vybrané trénovací množiny velmi dobře potvrdily předpoklady týkající se využití dané kartografické reprezentace na ZM 50 pro konkrétní klasifikovaný typ zástavby. U prvních dvou klasifikovaných typů výrazně převažuje kategorie *Bloková zástavba* a *Neohraničený blok*. Částečný výskyt *Kombinované* kategorie u těchto klasifikovaných typů zástavby poukazuje na chybnou segmentaci, popřípadě chybné přiřazení kategorie zástavby na ZM 50 konkrétnímu shluku. Klasifikovaný typ *Malé a střední domy* je dle očekávání tvořen všemi kategoriemi zástavby na ZM 50. U klasifikovaného typu *Industriální zástavba* byly předpoklady opět potvrzeny. Nejvíce shluků budov tohoto klasifikovaného typu je na ZM 50 znázorněno pomocí *Kombinované kategorie* a kategorie *Neohraničený blok*. V případě *Liniové zástavby* jsou shluky nejčastěji na ZM 50 zobrazovány jako *Samostatná budova* a částečně též jako *Kombinovaná* kategorie. Klasifikovaný typ *Samota* je poté dle očekávání tvořen pouze typem *Samostatná budova*.



Graf 14: Analýza závislosti mezi kategorií zástavby a klasifikovaným typem
Zdroj: *vlastní zpracování*

Hodnoty v dalších třech histogramech (graf 14) již vycházejí ze všech cca 170 000 shluků budov. Nejvíce se rozložení vycházejícího ze shluků ručně vybrané trénovací množiny blíží klasifikace pomocí neuronové sítě. Tato nejvyšší shoda je zajímavá

i z pohledu, že v případě ověřování přesnosti klasifikace na datech ručně vybrané trénovací množiny, dosahovala neuronová síť nejnižší přesnosti klasifikace ze všech tří klasifikátorů. Velice podobné rozložení četností bylo též naměřeno v případě logistické regrese. Oproti rozložení četností, zjištěných na základě dat z ručně vybrané trénovací množiny, se rozložení četností z logistické regrese a neuronové sítě nejvíce liší u klasifikovaného typu *Blok*. Oba klasifikátory zařadily do tohoto klasifikovaného typu mnohem více shluků, které jsou na ZM 50 znázorněny jako kategorie *Neohraničený blok*. Takto výrazný rozdíl může být způsoben nereprezentativním výběrem dat v ručně vybrané trénovací množině, spíše než chybou klasifikátoru. Naopak jako chybu klasifikátoru lze označit výskyt kategorie *Samostatná budova* u klasifikovaného typu *Blok* a *Blok s vnitroblokem*. Vyšší výskyt *Kombinované* kategorie u prvních dvou klasifikovaných typů lze přičíst nejspíše chybám při segmentaci. V případě klasifikovaného typu *Malé a střední domy* se rozložení četnosti oproti ručně vybrané trénovací množině opět částečně liší. Tento rozdíl může být opět částečně vysvětlen pomocí nereprezentativního výběru dat v ručně vybrané trénovací množině. Částečná odlišnost je nejspíše způsobena i chybami klasifikátorů, neboť klasifikovaný typ *Malé a střední domy* patřil mezi typy s nejnižší přesností klasifikace. Rozložení četností u zbývajících třech klasifikovaných typů téměř přesně odpovídá rozložení četností vycházejícího z dat ručně vybrané trénovací množiny. Výraznější rozdíl je patrný pouze v případě 7 % shluků budov z klasifikovaného typu *Samota*, jejichž typem kartografické reprezentace na ZM 50 je *Neohraničený blok*. Z definice klasifikovaného typu *Samota* vyplývá, že těchto 7 % shluků budov bylo nejspíše klasifikátorem chybně zařazeno. Část z těchto 7 % shluků mohla být též chybně označena algoritmem pro přiřazení kategorie zástavby na ZM 50 jednotlivým shlukům.

Výrazně odlišné rozložení četností bylo naměřeno v případě klasifikátoru využívajících CNN u prvních dvou klasifikovaných typů: *Blok* a *Blok s vnitroblokem.* Oproti předchozím dvěma klasifikátorům, označil klasifikátor CNN mnohem více shluků budov, které jsou na ZM 50 reprezentovány typem *Neohraničený blok,* jako klasifikovaný typ *Blok.* V případě shluků budov klasifikovaných jako typ *Blok s vnitroblokem* je mnohem menší část tvořena shluky budov, jejichž kartografickou reprezentací je typ *Bloková zástavba.* Naopak je v tomto klasifikovaném typu vyšší zastoupení typu kartografické reprezentace *Samotná budova*. Rozložení četností u zbývajících klasifikovaných typů zástavby je v případě CNN téměř shodné jako v přídě klasifikátoru neuronových sítí.

6 Závěr

Tato diplomová práce se zabývá tématem klasifikace zástavby pro účely kartografické generalizace. Hlavním cílem bylo představit a otestovat algoritmus pro klasifikaci zástavby se zaměřením na generalizaci zástavby na ZM 50. Samotná generalizace již není předmětem této práce. Pro splnění hlavního cíle práce byly vytyčeny tří vedlejší cíle práce. Prvním vedlejším cílem práce bylo představit typologii zástavby pro účely klasifikace s důrazem na propojení jednotlivých typů zástavby a kartografických reprezentací využívaných pro znázornění zástavby na ZM 50. Druhým vedlejším cílem bylo vytvoření algoritmu pro segmentaci budov do shluků. Posledním vedlejším cílem práce bylo porovnat různé klasifikační algoritmy. Cílem bylo porovnat klasifikace na základě popisných charakteristik a na základě vizuálního vjemu a zároveň porovnat klasifikační algoritmy strojového učení oproti klasifikátorům založených na neuronových sítích.

V úvodu této práce byla popsána problematika automatizace generalizace zástavby a definovány představené cíle práce. V teoretické části práce (kapitola č. 2) byly představeny v současnosti využívané algoritmy pro segmentaci a klasifikaci zástavby, včetně podrobnějšího popisu fungování klasifikačních algoritmů vybraných pro účely této práce. V praktické části práci byla nejprve představena navržená typologie zástavby čítající celkem šest typů (kapitola č. 3). Následně byl představen popis a implementace vytvořeného algoritmu pro segmentaci budov do shluků (kapitola č. 4). Navržený algoritmus využívá kombinace segmentace pomocí již generalizované cestní sítě a metody DBSCAN. Pro účely klasifikace shluků budov bylo představeno celkem 22 popisných charakteristik. Pro účely klasifikátoru založeného na vizuálním vjemu byl představen algoritmus pro tvorbu rastrových podob shluků budov. Následně byl popsán proces klasifikace pomocí jednotlivých klasifikátorů. Pro účely klasifikace na základě popisných charakteristik byla vybrána logistická regrese a neuronová síť. Pro klasifikaci na základě vizuálního vjemu byla poté zvolena konvoluční neuronová síť (CNN). V kapitole č. 5 jsou vybrané klasifikátory porovnány nejprve na datech ručně vybrané trénovací množiny obsahující přibližně 530 shluků budov. Pro evaluaci klasifikace na větším množství dat bylo využito porovnání klasifikovaného typu a kartografické reprezentace použité pro jeho znázornění na ZM 50. Pomocí tohoto způsobu byly klasifikátory porovnány na rozsahu odpovídají 1/3 budov na území Česka.

Ze všech provedených analýz vyplynulo, že představené typy zástavby lze pomocí všech tří klasifikátorů v zástavbě identifikovat. Míra úspěšnosti určení se ukázala jako relativně proměnlivá, zejména v závislosti na konkrétním typu zástavby. Při porovnání klasifikátorů na ručně vybrané trénovací množině vykázaly jak klasifikátory využívající popisných charakteristik, tak klasifikátor využívající vizuální posouzení, velmi podobné výsledky. Z klasifikátorů využívající popisných charakteristik lepší výsledky vykázala logistická regrese. Nepatrně nižší přesnost klasifikátoru neuronových byla způsobena nejspíše příliš malou trénovací množinou. Pomocí CNN se podařilo zejména díky využití datové augmentace dosáhnout téměř stejných výsledků jako pomocí logistické regrese. Při porovnání klasifikátorů na všech vstupních datech se již částečně projevily rozdíly mezi klasifikátory využívající popisné charakteristiky a CNN. Na základě výsledků v kapitole 5.2 lze usuzovat, že klasifikátor CNN se dokázal hůře adaptovat na velice různorodé shluky budov nacházejících se ve všech vstupních datech. V tomto případě se popisné charakteristiky ukázaly jako lepší identifikátor jednotlivých typů zástavby. Nižší úspěšnost CNN lze přisuzovat zejména dvěma důvodům, které souvisí s ručně vybranou trénovací množinou. Prvním důvodem je fakt, že do ručně vybrané trénovací množiny byli vybíráni pouze typičtí představitelé konkrétních tříd. Druhým důvodem je pak nejspíše nedostatečný počet shluků budov v trénovací množině. Celkově lze říct, že všechny tři klasifikátory i přes rozdílné principy jejich fungování vykázaly velice podobné výsledky. U neuronových sítí využívající popisné charakteristiky a CNN lze však očekávat větší potenciál pro případné zlepšení, zejména při využití mnohonásobně větší trénovací množiny.

Pomocí provedených analýz v kapitole č. 5 byl též potvrzen předpoklad, že zejména mezi typy *Blok s vnitroblokem* a *Malé a střední domy* není přesně definovaná hranice, a tudíž bude docházet k částečné záměně těchto typů. I přes ne úplně jasně definované hranice jednotlivých typů zástavby se však pomocí navržených 22 charakteristik podařilo jednotlivé typy relativně dobře charakterizovat. Do budoucna by bylo vhodné rozšířit popisné charakteristiky i o popis vztahu jednotlivých shluků budov k sousedícím shlukům budov. Do rastrových podob shluků budov sloužících jako vstup do CNN bylo vhodné zapojit i některé další prvky jako je silniční síť, popřípadě informace o typu využití půdy či výšce budov. V rámci diplomové práce nebylo též testováno využití tzv. *Transfer learning* (Pan, Yang 2010) neboli využití již natrénované CNN a pouze její přizpůsobení pro klasifikaci vymezených typů zástavby. V úvahu též přichází možnost

využití speciálních architektur neuronových sítí, které dokáží kombinovat na vstupu jak popisné charakteristiky, tak obrazový vstup (Bojarski et al. 2016).

Částí celého navrženého postupu, která skýtá možnosti pro vylepšení, je proces segmentace budov do shluků. Navržený algoritmus dokázal zástavbu segmentovat velice dobře, avšak ne zcela bezchybně. Úspěšnost segmentačního algoritmu lze připočíst zejména využití již generalizované cestní sítě a dalších liniových prvků. To znamená, že nelze segmentační algoritmus označit jako zcela automatický, neboť předpokládá existenci těchto dat. Zároveň v této práci nebyla úspěšnost segmentace nijak kvantifikována. Kromě využití cestní sítě pro segmentaci by bylo možné zapojit do segmentace informace o jednotlivých budovách jako je hodnota atributu *Druh budovy* vrstvy *Budova, Blok budov* z dat ZABAGED. Tuto informaci by bylo možné též využít pro vstup do procesu klasifikace. V procesu segmentace by též bylo možné využít dalších informací o budově jako je například její výška. V úvahu též přichází nahrazení metody DBSCAN některým z grafových segmentačních algoritmů zmíněných v kapitole č. 2.2. Výhodou grafových algoritmů je skutečnost, že vlastnosti grafu lze využít i jako vstup do procesu klasifikace.

Na závěr lze konstatovat, že se podařilo z větší části splnit všechny vytyčené cíle práce. Avšak vzhledem k relativně proměnlivé úspěšnosti klasifikace jednotlivých typů zástavby nelze tuto práci bez dalšího vylepšení využít pro cíl, který byl primární motivací pro tvorbu této práce, tedy pro klasifikaci zástavby pro účely generalizace zástavby na ZM 50. I přes tento fakt, se v práci podařilo vytvořit komplexní algoritmus obsahující jak proces segmentace, tak proces samotné klasifikace včetně několika nových přístupů k řešení určitých problémů. Zejména využití CNN pro klasifikaci zástavby lze částečně označit za inovativní. Jako významnější dílčí výsledek lze zmínit též algoritmus pro přiřazení podoby kartografické reprezentace zástavby na ZM 50 příslušným shlukům. Tento algoritmus by bylo teoreticky možné, kromě jeho původního cíle, využít například pro hodnocení kvality generalizace, resp. hledání chyb či nejednotností generalizace zástavby na ZM 50. Tato práce též může sloužit jako podklad pro další navazující studie, jejichž cílem bude již samotná generalizace jednotlivých navržených typů zástavby.

Zdroje

Literatura

ALASADI, S. A., BHAYA, W. S. (2017): *Review of data preprocessing techniques in data mining*. Journal of Engineering and Applied Sciences, 16, 12, 4102–4107.

ANDERS, K.-H., SESTER, M., FRITSCH, D. (1999): *Analysis of Settlement Structures by Graph-Based Clustering*. In: Bückner, J. (ed.): SMATI 99: Semantic Modelling for the Acquisition of Topographic Information from Images and Maps. Munich, 41–49.

AROWOLO, M. O., ABDULSALAM, S. O., SAHEED, Y. K., SALAWU, M. D. (2016): A Feature Selection Based on One-Way-Anova for Microarray *Data Classification*. Al-Hikmah Journal of Pure & Applied Sciences, December 2016, 3, 1–6.

BASARANER, M., SELCUK, M. (2008): *A Structure Recognition Technique in Contextual Generalisation of Buildings and Built-up Areas*. The Cartographic journal, 4, 45, 274–285.

BELEITES, C., BAUMGARTNER, R., BOWMAN, C., SOMORJAI, R., STEINER, G., SALZER, R., SOWA, M. G. (2005): *Variance reduction in estimating classification error using sparse datasets*. Chemometrics and Intelligent Laboratory Systems, 1–2, 79, 91–100.

BERGSTRA, J., BENGIO, Y. (2012): *Random Search for Hyper-Parameter Optimization*. Journal of Machine Learning Research, 13, 281–305.

BISHOP, C. M. (2006): *Pattern Recogniton and Machine Learning*. Springer-Verlag New York, New York, NY.

BREIMAN, L. (2001): *Random forests.* Machine Learning, 1, 45, 5–32.

CAWLEY, G. C., TALBOT, N. L. C. (2010): *On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation*. Journal of Machine Learning Research, 11, 2079–2107.

CETINKAYA, S., BASARANER, M., BURGHARDT, D. (2015): *Proximity-based grouping of buildings in urban blocks: a comparison of four algorithms*. Geocarto International, 6, 30, 618–632.

CHANDRASHEKAR, G., SAHIN, F. (2014): *A survey on feature selection methods*. Computers and Electrical Engineering, 1, 40, 16–28.

CHRISTOPHE, S., RUAS, A. (2002): *Detecting Building Alignments for Generalisation Purposes*. In: Advances in Spatial Data Handling. Springer Berlin Heidelberg, Berlin, Heidelberg, 419–432.

CORTES, C., VAPNIK, V. (1995): *Support-Vector Networks*. Machine Learning, 3, 20, 273–297.

DO, L. N. N., TAHERIFAR, N., VU, H. L. (2019): *Survey of neural network-based models for short-term traffic state prediction*. WIREs Data Mining and Knowledge Discovery, 1, 9.

DOBROVOLNÝ, R. (1998*): Dálkový průzkum Země , Digitální zpracování obrazu*. Masarykova univerzita, Přírodovědecká fakulta, Brno.

DU, S., LUO, L., CAO, K., SHU, M. (2016): *Extracting building patterns with multilevel graph partition and building grouping*. ISPRS Journal of Photogrammetry and Remote Sensing, 122, 81–96.

ELSSIED, N. O. F., IBRAHIM, O., OSMAN, A. H. (2014): *A novel feature selection based on one-way ANOVA F-test for e-mail spam classification*. Research Journal of Applied Sciences, Engineering and Technology, 3, 7, 625–638.

ESTER, M., KRIEGEL, H.-P., SANDER, J., XU, X. (1996): *A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. AAAI Press, Portland, OR, USA, 226–231.

EVANS, J. D. (1996): Straightforward statistics for the behavioral sciences. Thomson Brooks/Cole Publishing Co, Belmont, CA, US.

FUKUSHIMA, K. (1980): *Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position*. Biological Cybernetics, 4, 36, 193–202.

GOLDBERG, D. E., HOLLAND, J. H. (1988): *Genetic Algorithms and Machine Learning*. Machine Learning, 2, 3, 95–99.

GOTTSTEIN, O. (2019): *Generalizace zástavby s využitím typifikace*. Diplomová práce. Univerzita Karlova, Přírodovědecká fakulta, Katedra aplikované geoinformatiky a kartografie.

GREDELL, D. A., SCHROEDER, A. R., BELK, K. E., BROECKLING, C. D., HEUBERGER, A. L., KIM, S. Y., KING, D. A., SHACKELFORD, S. D., SHARP, J. L., WHEELER, T. L., WOERNER, D. R., PRENNI, J. E. (2019): *Comparison of Machine Learning Algorithms for Predictive Modeling of Beef Attributes Using Rapid Evaporative Ionization Mass Spectrometry (REIMS) Data*. Scientific Reports, 1, 9, 1–9.

GUO, Q., WU, W., MASSART, D. L., BOUCON, C., DE JONG, S. (2002): *Feature selection in principal component analysis of analytical data*. Chemometrics and Intelligent Laboratory Systems, 1–2, 61, 123–132.

GUYON, I., ANDRÉ, E. (2003): *An Introduction to Variable and Feature Selection*. Journal of Machine Learning Research, 3, 1157–1182.

GUYON, I., WESTON, J., BARNHILL, S., VAPNIK, V. (2002): *Gene selection for cancer classification using support vector machines*. Machine Learning, 1–3, 46, 389–422.

HE, K., ZHANG, X., REN, S., SUN, J. (2016): *Deep Residual Learning for Image Recognition*. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, 770–778.

HE, X., ZHANG, X., XIN, Q. (2018): *Recognition of building group patterns in topographic maps based on graph partitioning and random forest*. ISPRS Journal of Photogrammetry and Remote Sensing, 136, 26–40.

HECHT, R., HEROLD, H., MEINEL, G., BUCHROITHNER, M. (2013): *Automatic Derivation of Urban Structure Types from Topographic Maps by Means of Image Analysis and Machine Learning*. In: 26th International Cartographic Conference. International Cartographic Association, 18.

HORNIK, K., STINCHCOMBE, M., WHITE, H. (1989): *Multilayer feedforward networks are universal approximators*. Neural Networks, 5, 2, 359–366.

HUANG, J., LI, Y. F., XIE, M. (2015): *An empirical analysis of data preprocessing for machine learning-based software cost estimation*. Information and Software Technology, 67, 108–127.

HUTTER, F., HOOS, H. H., LEYTON-BROWN, K. (2011): *Sequential model-based optimization for general algorithm configuration*. In: Coello, C. A. (ed.): Learning and Intelligent Optimization. LION 2011. Springer, Berlin, Heidelberg, 507–523.

HUTTER, F., LÜCKE, J., SCHMIDT-THIEME, L. (2015): *Beyond Manual Tuning of Hyperparameters*. KI - Künstliche Intelligenz, 4, 29, 329–337.

IOFFE, S., SZEGEDY, C. (2015): *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. In: 32nd International Conference on Machine Learning, ICML 2015. International Machine Learning Society (IMLS), 448–456.

KARYPIS, G., KUMAR, V. (1998): *A fast and high quality multilevel scheme for partitioning irregular graphs*. SIAM Journal of Scientific Computing, 1, 20, 359–392.

KILPELAINEN, T. (1995): *Updating multiple representation geodata bases by incremental generalization*. Geo-Informations-Systeme, 4, 8, 13–18.

KINGMA, D. P., BA, J. L. (2015): *Adam: A method for stochastic optimization*. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. International Conference on Learning Representations, ICLR.

KOHONEN, T. (1982): *Self-organized formation of topologically correct feature maps*. Biological Cybernetics, 1, 43, 59–69.

KONONENKO, I. (1994): *Estimating attributes: Analysis and extensions of RELIEF*. In: European Conference on Machine Learning. Springer, Berlin, Heidelberg, 171–182.

KUHN, M., JOHNSON, K. (2013): *Applied Predictive Modeling*. Springer New York, New York, NY.

LECUN, Y., BENGIO, Y. (1998): *Convolutional Networks for Images, Speech, and Time Series.* In: The Handbook of Brain Theory and Neural Networks. MIT Press, Cambridge, MA, USA, 255–258.

LECUN, Y., HAFFNER, P., BOTTOU, L., BENGIO, Y. (1999): *Object recognition with gradient-based learning*. In: Shape, Contour and Grouping in Computer Vision. Springer-Verlag Berlin Heidelberg, 319–345.

LEE, S. I., LEE, H., ABBEEL, P., NG, A. Y. (2006): *Efficient L 1 regularized logistic regression*. In: Proceedings of the National Conference on Artificial Intelligence. Boston, Massachusetts, USA, 401–408.

LI, Z., YAN, H., AI, T., CHEN, J. (2004): *Automated building generalization based on urban morphology and Gestalt theory*. International Journal of Geographical Information Science, 5, 18, 513–534.

LIQIANG, Z., HAO, D., DONG, C., ZHEN, W. (2013*): A spatial cognition-based urban building clustering approach and its applications*. International Journal of Geographical Information Science, 4, 27, 721–740.

LONG, D., ZHANG, S., ZHANG, Y. (2019): *Performance Prediction Based on Neural Architecture Features*. In: 2019 2nd China Symposium on Cognitive Computing and Hybrid Intelligence (CCHI). IEEE, Xi'an, China, 77–80.

MACEK, K. (2008): Pareto principle - data mining. Acta Polytechnica, 6, 48, 55–59.

MAĆKIEWICZ, A., RATAJCZAK, W. (1993): *Principal components analysis (PCA)*. Computers and Geosciences, 3, 19, 303–342.

MANNOR, S., JIN, X., HAN, J., JIN, X., HAN, J., JIN, X., HAN, J., ZHANG, X. (2011): *K-Means Clustering*. In: Encyclopedia of Machine Learning. Springer US, Boston, MA, 563–564.

MAZUROWSKI, M. A., HABAS, P. A., ZURADA, J. M., LO, J. Y., BAKER, J. A., TOURASSI, G. D. (2008): *Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance*. Neural Networks, 2–3, 21, 427–436.

MEINEL, G., HECHT, R., HEROLD, H. (2009): *Analyzing building stock using topographic maps and GIS*. Building Research and Information, 5–6, 37, 468–482.

MENARD, S. (2002): *Applied Logistic Regression Analysis*. SAGE Publications, Inc., Thousand Oaks, California, USA.

MUNKHDALAI, L., MUNKHDALAI, T., NAMSRAI, O.-E., LEE, J., RYU, K. (2019): *An Empirical Comparison of Machine-Learning Methods on Bank Client Credit Assessments*. Sustainability, 3, 11, 699.

PAN, S. J., YANG, Q. (2010): *A Survey on Transfer Learning*. IEEE Transactions on Knowledge and Data Engineering, 10, 22, 1345–1359.

PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M., DUCHESNAY, E. (2011): *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.

REGNAULD, N. (2001): Contextual *Building Typification in Automated Map Generalization*. Algorithmica, 2, 30, 312–333.

ŘEHÁKOVÁ, B. (2000): *Nebojte se logistické regrese*. Sociologický časopis / Czech Sociological Review, 4, 36, 475–492.

REN, Y., BAI, G. (2011): *New neural network response surface methods for reliability analysis*. Chinese Journal of Aeronautics, 1, 24, 25–31.

SNOEK, J., LAROCHELLE, H., ADAMS, R. P. (2012): *Practical Bayesian Optimization of Machine Learning Algorithms*. In: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2. Curran Associates Inc., Lake Tahoe, Nevada, 2951–2959.

SONG, F., GUO, Z., MEI, D. (2010): *Feature selection using principal component analysis*. In: 2010 International Conference on System Science, Engineering Design and Manufacturing Informatization, ICSEM 2010. 27–30.

SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SALAKHUTDINOV, R. (2014): *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. Journal of Machine Learning Research, 15, 1929–1958.

STEINIGER, S., BURGHARDT, D., WEIBEL, R., LANGE, T., BURGHARDT, D., WEIBEL, R. (2008): *An approach for the classification of urban building structures based on discriminant analysis techniques*. Transactions in GIS, 1, 12, 31–59.

SUN, C., SHRIVASTAVA, A., SINGH, S., GUPTA, A. (2017): *Revisiting Unreasonable Effectiveness of Data in Deep Learning Era*. In: Proceedings of the IEEE International Conference on Computer Vision. Institute of Electrical and Electronics Engineers Inc., 843–852.

SUTSKEVER, I., VINYALS, O., LE, Q. V. (2014): *Sequence to Sequence Learning with Neural Networks*. In: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2. MIT Press, Montreal, Canada, 3104–3112.

VANDERHAEGEN, S., CANTERS, F. (2010): *Developing urban metrics to describe the morphology of urban areas at block level*. In: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. 6.

VERWAEREN, J., VAN DER WEEËN, P., DE BAETS, B. (2015): *A search grid for parameter optimization as a byproduct of model sensitivity analysis*. Applied Mathematics and Computation, 261, 8–27.

WANG, Z., HUTTER, F., ZOGHI, M., MATHESON, D., DE FREITAS, N. (2016): *Bayesian optimization in a billion dimensions via random embeddings*. Journal of Artificial Intelligence Research, 55, 361–367.

WISTUBA, M., RAWAT, A., PEDAPATI, T. (2019): *A Survey on Neural Architecture Search*. Journal of Machine Learning Research, 20, 1–21.

WU, J., CHEN, X. Y., ZHANG, H., XIONG, L. D., LEI, H., DENG, S. H. (2019): *Hyperparameter optimization for machine learning models based on Bayesian optimization*. Journal of Electronic Science and Technology, 1, 17, 26–40.

YAN, H., WEIBEL, R., YANG, B. (2008): *A Multi-parameter Approach to Automated Building Grouping and Generalization*. GeoInformatica, 1, 12, 73–89.

YAN, X., AI, T., YANG, M., YIN, H. (2019): *A graph convolutional neural network for classification of building patterns using spatial vector data*. ISPRS Journal of Photogrammetry and Remote Sensing, February, 150, 259–273.

ZHANG, X., AI, T., STOTER, J. (2012*): Characterization and detection of building patterns in cartographic data: Two algorithms*. Lecture Notes in Geoinformation and Cartography, 38, 93–107.

ZHANG, X., AI, T., STOTER, J., MOLENAAR, M. K. M., KRAAK, M.-J., MOLENAAR, M. K. M. (2013): *Building pattern recognition in topographic data : examples on collinear and curvilinear alignments*. GeoInformatica, 1, 17, 1–33.

ZOPH, B., VASUDEVAN, V., SHLENS, J., LE, Q. V. (2018*): Learning Transferable Architectures for Scalable Image Recognition*. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 8697–8710.

ZVÁRA, K., ANDĚL, J., MARTINKOVÁ, P. (2013): *Základy statistiky v prostředí R*. Karolinum, Praha.

Internetové a další zdroje

ABADI, M., BARHAM, P., CHEN, J., CHEN, Z., DAVIS, A., DEAN, J., DEVIN, M., GHEMAWAT, S., IRVING, G., ISARD, M., KUDLUR, M., LEVENBERG, J., MONGA, R., MOORE, S., MURRAY, D. G., STEINER, B., TUCKER, P., VASUDEVAN, V., WARDEN, P., WICKE, M., YU, Y., ZHENG, X., BRAIN, G., OSDI, I., BARHAM, P., CHEN, J., CHEN, Z., DAVIS, A., DEAN, J., DEVIN, M., GHEMAWAT, S., IRVING, G., ISARD, M., KUDLUR, M., LEVENBERG, J., MONGA, R., MOORE, S., MURRAY, D. G., STEINER, B., TUCKER, P., VASUDEVAN, V., WARDEN, P., WICKE, M., YU, Y., ZHENG, X. (2016): *TensorFlow : A System for Large-Scale Machine Learning*, https://tensorflow.org /(10. 12. 2019).

AKIBA, T., SANO, S., YANASE, T., OHTA, T., KOYAMA, M. (2019): *Optuna: A Next-generation Hyperparameter Optimization Framework*, https://optuna.org/ (1. 2. 2020).

JACKEL, L. D., MONFORT, M., MULLER, U., ZHANG, J., ZHANG, X., ZHAO, J., ZIEBA, K. (2016): *End to End Learning for Self-Driving Cars*, http://arxiv.org/abs/1604.07316 (4. 1. 2020).

BROWNLEE, J. (2019): *Machine learing mystery*, https://machinelearningmastery.com/ (20. 2. 2020).

COOK, A. (2017): *Global Average Pooling Layers for Object Localization*, https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/ (31. 3. 2020).

ČÚZK (2018): *Katalog objektů ZABAGED*, http://geoportal.cuzk.cz/Dokumenty/ZABAGED_katalog/index.html (12. 8. 2019).

ČÚZK (2019a): *Terminilogický slovník zeměměřictví a katastru nemovitostí*, https://www.vugtk.cz/slovnik/termin.php?jazykova_verze=&tid=1143&l=mapa (28. 9. 2019).

DEEP AI (2018): *Neural Network*, https://deepai.org/machine-learning-glossary-and-terms/neural-network (29. 4. 2019).

DESHPANDE, A. (2016): *A Beginner's Guide To Understanding Convolutional Neural Networks*, https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/ (28. 1. 2019).

DI GREGORIO, F. (2020): *Psycopg*, https://pypi.org/project/psycopg2/ (12. 2. 2020).

ESCONTRELA, A. (2017): *Convolutional Neural Networks from the ground up*, https://towardsdatascience.com/convolutional-neural-networks-from-the-ground-up-c67bb41454e1 (31. 3. 2020).

ESRI (2020): *ArcPy*, https://pro.arcgis.com/en/pro-app/arcpy/get-started/what-is-arcpy-.htm (12. 2. 2020).

GOOGLE (2020): *Google Colab*, https://research.google.com/colaboratory/faq.html (12. 2. 2020).

HEATON, J. (2019): *Applications of Deep Neural Networks*, https://github.com/jeffheaton/t81_558_deep_learning/blob/master/README.md (28. 1. 2019).

KRUT, P. (2019): *Convolutional Neural Networks — A Beginner's Guide*, https://towardsdatascience.com/convolution-neural-networks-a-beginners-guide-implementing-a-mnist-hand-written-digit-8aa60330d022 (5. 2. 2020).

LIU, E. (2019): *Model selection*, http://ethen8181.github.io/machine-learning/model_selection/model_selection.html (1. 2. 2020).

MICROSOFT (2020): GPUs vs CPUs for deployment of deep learning models, https://azure.microsoft.com/cs-cz/blog/gpus-vs-cpus-for-deployment-of-deep-learning-models/ (1. 2. 2020).

NG, A. (2019): *Deep Learning*, http://ufldl.stanford.edu/ (4. 2. 2020).

PIATETSKY-SHAPIRO, G. (2020): *KDnuggets - Regularization in Logistic Regression*, https://www.kdnuggets.com/2016/06/regularization-logistic-regression.html (20. 2. 2020).

What is Python? (2020): https://www.python.org/doc/essays/blurb/ (6. 3. 2020).

RAMACHANDRAN, P., ZOPH, B., LE, Q. V. (2017): *Searching for Activation Functions*, http://arxiv.org/abs/1710.05941 (29. 3. 2020).

RASCHKA, S. (2014): *Gradient Descent*, http://rasbt.github.io/mlxtend/user_guide/general_concepts/gradient-optimization/#references (5. 3. 2020).

SCIKIT-LEARN (2018): User Guide, https://scikit-learn.org/stable/user_guide.html (1. 12. 2019).

SZEGEDY, C., VANHOUCKE, V., IOFFE, S., SHLENS, J., WOJNA, Z. (2015): *Rethinking the Inception Architecture for Computer Vision*, http://arxiv.org/abs/1512.00567 (5. 3. 2020).

TAYLOR, L., NITSCHKE, G. (2017): *Improving Deep Learning using Generic Data Augmentation*, http://arxiv.org/abs/1708.06020 (9. 3. 2020).

TIŠNOVSKÝ, P. (2018): *Seriál Torch: framework pro strojové učení*, https://www.root.cz/serialy/torch-framework-pro-strojove-uceni/ (29. 2. 2019).

PEREZ, L., WANG, J. (2017): *The Effectiveness of Data Augmentation in Image Classification using Deep Learning*, http://arxiv.org/abs/1712.04621 (9. 3. 2020).

WARDEN, P. (2017): *How many images do you need to train a neural network*? , https://petewarden.com/2017/12/14/how-many-images-do-you-need-to-train-a-neural-network/ (9. 3. 2020).

WIKIPEDIA (2020): *Logistic regresion*, https://en.wikipedia.org/wiki/Logistic_regression (10. 2. 2020).

Datové zdroje

ČÚZK (2020a): *Data50 – mapový podklad,* https://geoportal.cuzk.cz/(S(l1o3tnpqakoawytrida252b2))/Default.aspx?mode=TextMeta&side=mapy_data50&text=dSady_mapyData50&head_tab=sekce-02-gp&menu=2290 (12. 2. 2020).

ČÚZK (2020b): *DMP 1G - ImageServer*, http://ags.cuzk.cz/arcgis2/rest/services/dmp1g/ImageServer (12. 2. 2020).

ČÚZK (2020c): *DMR 5G - ImageServer*, http://ags.cuzk.cz/arcgis2/rest/services/dmr5g/ImageServer (12. 2. 2020).

ČÚZK (2019d): *ZABAGED - polohopis*, https://geoportal.cuzk.cz/(S(whdztwgl0s5zi5etc4u2g21l))/Default.aspx?lng=CZ&mode=TextMeta&side=zabaged&metadataID=CZ-CUZK-ZABAGED-VP&mapid=8&menu=241 (12. 2. 2020).

ČÚZK (2020e): *Základní mapa České republiky 1 : 50 000*, https://geoportal.cuzk.cz/(S(dzakwd0ga0slkdlyqfwvmj04))/Default.aspx?mode=TextMeta&side=mapy50&text=dsady_mapy50&menu=225 (25. 3. 2020).

Přílohy

Vzhledem k formátu příloh jsou veškeré přílohy dostupné pouze v elektronické podobě. Zde je uveden pouze jejich krátký popis.

Příloha 1: Skripty

Celá navržená metoda je implementována pomocí čtyř skriptů v jazyce *Python 3*. První skript (*segmentation_stats.py*) slouží pro segmentaci budov do shluků (kapitola 4.3) a též pro výpočet popisných charakteristik jednotlivých shluků budov (kapitola 4.4). Skript (*create_rasters.py*) slouží pro tvorbu rastrových podob shluků (kapitola 4.5) pro účely CNN. Třetí skript (*classification.ipynb*) obsahuje veškeré výpočty spojené s procesem klasifikace (kapitola 4.7) a porovnání jednotlivých klasifikátorů (kapitola 5). Z důvodu využití vývojového prostředí *Google Colab* je poslední skript uložen ve formátu „*ipynb*“ (*Jupyter notebook*). Poslední skript (*ZM_blocks.py*) obsahuje implementaci metody pro přiřazení kategorie zástavby na ZM 50 jednotlivým shlukům budov (kapitola 5.2). Veškeré přiložené skripty byly sepsány pouze pro experimentální účely a dokazují realizovatelnost představené metody, a tudíž nejsou zamýšlené jako produkční verze pro přímé použití dalšími uživateli.

Příloha 2: Polygonová vrstva shluků budov

Výsledná polygonová vrstva (*building_clusters.shp*) obsahuje dissolvované polygony vstupních budov podle shluků budov, do kterých náleží (jeden řádek v atributové tabulce představuje jeden shluk budov). Pro každý klasifikátor obsahuje atributová tabulka celkem 7 atributů (tabulka č. 21). Jeden atribut představuje nejpravděpodobnější typ zástavby daného shluku budov podle daného klasifikátoru. Zbylých šest atributů představuje hodnoty pravděpodobnosti zařazení daného shluku budov do jednotlivých typů zástavby. Dále atributová tabulka obsahuje 22 spočtených popisných charakteristik (kapitola 4.4). Dále atributová tabulka obsahuje atribut, který informuje o kategorii zástavby na ZM 50. Atributová tabulka též obsahuje informaci, jaké shluky byly vybrány do trénovací množiny. Každý shluk budov je též určen pomocí unikátního identifikátoru. Převod mezi aliasy zanesenými v atributové tabulce a celými názvy atributů je uveden v tabulce č. 21 na straně 128. Tabulka č. 22 na straně 129 poté obsahuje číselník jednotlivých typů/kategorií zástavby.

Alias	Celý název
cluster_ID	Jedinečný identifikátor shluku
build_area	Součet plochy budov
mean_b_ar	Průměrná plocha budovy
build_len	Součet obvodů budov
mean_b_len	Průměrný obvod budovy
build_c	Počet budov
Con_length	Obvod konvexní obálky
Con_area	Plocha konvexní obálky
RBW_width	Šířka opsaného obdélníku s minimální šířkou
RBW_length	Délka opsaného obdélníku s minimální šířkou
RBW_area	Plocha opsaného obdélníku s minimální šířkou
RBA_width	Šířka opsaného obdélníku s minimální plochou
RBA_length	Délka opsaného obdélníku s minimální plochou
RBA_area	Plocha opsaného obdélníku s minimální plochou
bw_index	Poměr plochy budov ku ploše konvexní obálky
shape_ind	Poměr šířky a délky opsaného obdélníku s minimální šířkou
std_area	Směrodatná odchylka plochy budov
std_orient	Směrodatná odchylka orientace budovy
road_dist	Průměrná vzdálenost k silnici
mean_NN	Průměrná vzdálenost k nejbližší budově
mean_KNN	Průměrná vzdálenost ke třem nejbližším budovám
mean_height	Průměrná výška budovy
ZM_block_ratio	Poměr plochy budov k ploše polygonu z první části segmentace

Alias	Celý název
LR_blok	Pravděpodobnost zařazení do typu Blok klasifikátorem LR
LR_vnitroblok	Pravděpodobnost zařazení do typu Blok s vnitroblokem klasifikátorem LR
LR_male_stredni	Pravděpodobnost zařazení do typu Malé a střední domy klasifikátorem LR
LR_industrial	Pravděpodobnost zařazení do typu Industriální zástavba klasifikátorem LR
LR_linie	Pravděpodobnost zařazení do typu Liniová zástavba klasifikátorem LR
LR_samota	Pravděpodobnost zařazení do typu Samota klasifikátorem LR
LR	Nejpravděpodobnější typ zástavby určení klasifikátorem LR
NN_blok	Pravděpodobnost zařazení do typu Blok klasifikátorem NN
NN_vnitroblok	Pravděpodobnost zařazení do typu Blok s vnitroblokem klasifikátorem NN
NN_male_stredni	Pravděpodobnost zařazení do typu Malé a střední domy klasifikátorem NN
NN_industrial	Pravděpodobnost zařazení do typu Industriální zástavba klasifikátorem NN
NN_linie	Pravděpodobnost zařazení do typu Liniová zástavba klasifikátorem NN
NN_samota	Pravděpodobnost zařazení do typu Samota klasifikátorem NN
NN	Nejpravděpodobnější typ zástavby určení klasifikátorem neuronových sítí
CNN_blok	Pravděpodobnost zařazení do typu Blok klasifikátorem CNN
CNN_vnitroblok	Pravděpodobnost zařazení do typu Blok s vnitroblokem klasifikátorem CNN
CNN_male_stredni	Pravděpodobnost zařazení do typu Malé a střední domy klasifikátorem CNN
CNN_industrial	Pravděpodobnost zařazení do typu Industriální zástavba klasifikátorem CNN
CNN_linie	Pravděpodobnost zařazení do typu Liniová zástavba klasifikátorem CNN
CNN_samota	Pravděpodobnost zařazení do typu Samota klasifikátorem CNN
CNN	Nejpravděpodobnější typ zástavby určení klasifikátorem konvolučních neuronových sítí
ZM_TYPE	Kategorie zástavby na ZM 50
Typ_rucni	Shluky ručně vybrané trénovací množiny

Tabulka 21: Převodní tabulka mezi aliasy charakteristik a jejich celými názvy
***Zdroj**: vlastní zpracování*

Příloha 3: Naučené klasifikátory

K diplomové práci jsou též přiloženy naučené klasifikátory, které byly použity v kapitole č. 5.2. Klasifikátor logistické regrese (*log_res.joblib*) byl implementován pomocí knihovny

Scikit-learn a proto je uložen ve formátu „*joblib*“. Klasifikátory založené na neuronových sítích (*NN.h5, CNN.h5*) jsou poté uloženy ve formátu „*HDF5*“. V případě využití těchto naučených klasifikátorů je nutné zachovat stejné rozměry vstupních dat, jako byly využity v této diplomové práci. Pro klasifikátory využívající popisných charakteristik to znamená použití stejného počtu popisných charakteristik, včetně zachování jejich pořadí. Charakteristiky využité v kapitole 5.2 včetně jejich pořadí jsou uvedeny v tabulce č. 23 na straně 129. Oba klasifikátory též na vstupu očekávají normalizované hodnoty. Pro klasifikátor CNN jsou na vstupu očekávány jednopásmové rastry o rozměrech 256x256 pixelů.

Typ_rucni/LR/NN/CNN	
hodnota	**Typ zástavby**
-1	shluk nevybrán do trénovací množiny
0	Blok
1	Blok s vnitroblokem
2	Malé a střední budovy
3	Industriální zástavba
4	Liniová zástavba
5	Samota

ZM_TYPE	
hodnota	**Kategorie zástavby na ZM 50**
1	Bloková zástavba
2	Neohraničený blok
3	Samostatná budova
23	Kombinovaný typ

Tabulka 22: Převod mezi číselným označením typů/kategorií a celým názvem
***Zdroj**: vlastní zpracování*

pořadí	LR	NN
1	build_area	build_area
2	mean_b_ar	mean_b_ar
3	build_len	build_len
4	mean_b_len	mean_b_len
5	Con_length	build_c
6	bw_index	Con_length
7	shape_ind	Con_area
8	std_orient	RBW_width
9	road_dist	bw_index
10	mean_NN	shape_ind
11	mean_KNN	std_area
12	mean_height	std_orient
13	ZM_block_ratio	road_dist
14		mean_NN
15		mean_KNN
16		mean_height
17		ZM_block_ratio

Tabulka 23: Seznam a pořadí vybraných atributů pro klasifikátory LR A NN
***Zdroj**: vlastní zpracování*